Computational Models for the Automatic Learning and Recognition of Irish Sign Language

by

Daniel Kelly BSc.

NUI MAYNOOTH

Ollscoil na hÉireann Má Nuad

A thesis presented in fulfilment of the requirements for the Degree of

Doctor of Philosophy

Supervisor: Dr. Charles Markham, Mr. John Mc Donald

Department of Computer Science

National University of Ireland, Maynooth

Maynooth, Co.Kildare, Ireland

November, 2010

For Mary, Ollie and Ramona

Declaration

This thesis has not been submitted in whole or in part to this or any other university for any other degree and is, except where otherwise stated, the original work of the author.

Signed: _____

Daniel Kelly BSc.

Acknowledgments

It is a pleasure to thank those who made this thesis possible. I would like to thank my supervisors Dr Charles Markham and John Mc Donald for their continued support throughout my years working on this thesis. I would also like to thank my fellow colleagues in the computer science department and particularly Jane and Ron for their help on this thesis.

The biggest thanks has to go to my parents and Ramona. Without the support of my parents, this work leading to this thesis would not have even begun. Without Ramona's support, encouragement and patience, this thesis would not have been completed.

This thesis would not have been possible without the financial support of the Embark Initiative of the Irish Research Council for Science, Engineering, and Technology.

Contents

List of Figures

List of Tables

List of Acronyms

AAM Active Appearance Models. 65

ASL American Sign Language. 58, 61, 64, 65, 71, 159

ASM Active Shape Models. 148

AUC Area Under the Curve. 97, 99–101, 104, 105, 109, 114, 153–155, 157–160, 162, 165, 182, 243, 265

CRF Conditional Random Field. 40, 65, 116, 141, 142, 144–146, 149–155, 157–160, 162, 167, 181, 182, 244, 245

CSL Chinese Sign Language. 63

EMG Electromyogram. 50, 60

GT-HMM Gesture Threshold Hidden Markov Model. 116, 124, 135–137, 141, 146, 150, 152–155, 157–159, 162, 163, 165–170, 173, 174, 176, 181–183, 185, 206, 212, 231, 232, 239, 244, 245

HCI Human Computer Interface. 49, 72

HCRF Hidden Conditional Random Field. 40, 116, 143–146, 149, 150, 152, 153, 155, 166, 167, 181, 182, 244

HMM Hidden Markov Model. 35, 40, 58, 60, 62–64, 66–68, 71, 74, 116–121, 123–128, 133–136, 139, 141, 142, 144, 149–155, 157, 159, 160, 162, 163, 166, 167, 181–183, 212, 233, 244, 245, 249, 251, 253

IOHMM Input/Output Hidden Markov Model. 163

ISL Irish Sign Language. 33, 92, 93, 96, 97, 99–101, 104, 105, 108, 109, 114, 147, 184, 188, 222, 238, 243

LDCRF Latent Dynamic Conditional Random Field. 40, 116, 144–146, 149, 150, 152–155, 158, 159, 161, 162, 165–170, 173, 174, 181–183, 244, 245

MIL Multiple Instance Learning. 35, 41, 74, 76, 187, 193–195, 223, 240

MSER Maximally Stable Extremal Region. 55

PCA Principal Component Analysis. 54, 69, 89, 90, 129

PDF Probability Density Function. 119

RBF Radial Basis Function. 106, 107, 243, 260

ROC Receiver Operating Characteristics. 97, 99, 100, 102–104, 108, 109, 114, 153, 157, 160, 165, 168, 243, 263–265

SVM Support Vector Machine. 105, 106, 108, 111, 114, 206, 218, 231, 239, 243, 255, 259, 261

T-HMM Threshold Hidden Markov Model. 150, 152–155, 157, 159, 162, 165–167, 244

TMM Transition Movement Models. 62

Glossary of Mathmatical Notation

Chapter 3: Hand Posture Recognition Glossary

C	Contour extracted from binary image of hand. 85, 97–99, 101, 105, 107, 108, 189, 192, 219
$\ell_{\varphi\theta}$	Size Function generated from measuring function φ_θ. 86, 89
Γ_φ	Set of Size Functions $\Gamma_\varphi = \{\ell_{\varphi 1}, \ell_{\varphi 2}, ..., \ell_{\varphi N_\Theta}\}$. 86, 97
$G_{\varphi \leq y}$	Subgraph of G determined by points with $\varphi(p) \leq y$. 85, 103
I	Set of Hu Moments $I = \{I_1, I_2, I_3, I_4, I_5, I_6, I_7\}$. 84, 105–108, 218, 219
M_θ	EigenVector Matrix representing an ordered set of eigenvectors calculated from size function $\ell_{\varphi\theta}$. 90, 97
N_Θ	Total number of rotations used in the family of measuring function indexed by θ. 86, 103–105
N	Width and Height of Size Function. 89, 90, 97, 103–105
P	Number of Principal Components. 97, 98, 103–105

φ_θ	Measuring function which rotates each point of G about the centre of gravity by an angle of θ and measures the distance between the horizontal axis and each point p. 86
W_θ	EigenValue Matrix representing an ordered set of eigenvalues calculated from size function $\ell_{\varphi\theta}$. 90, 98
X_θ	Matrix notation for Size function $\ell_{\varphi\theta}$. 89
Y	Set of hand posture classes $Y = \{\alpha_1, \alpha_2, ..., \alpha_Y\}$. 105
α_y	Hand posture class y. 105, 107, 219
cv_y^{hu}	Cross Validation Accuracy achieved by SVM_y^{hu}. 107
cv_y^{sf}	Cross Validation Accuracy achieved by SVM_y^{sf}. 107
d^{Hu}	Distance Function to calculate Hu Moment distance between two hand contours. 192
d^{SF}	Distance Function to calculate weighted eigenspace size function distance between two hand contours. 99, 192
ℓ_φ	Size Function induced by Measuring function φ. 85
μ_y^{hu}	Hu Moment classifier weight. 107, 109
μ_y^{sf}	Weighted Eigenspace size function classifier weight. 107, 109
SVM_y^{hu}	Support Vector Machine which classifies Hu Moments for posture class y. 105, 107, 218
SVM_y^{sf}	Support Vector Machine which classifies Size Functions for posture class y. 105, 107, 218
φ	Measuring function. 85
ϖ_θ	Weighting factor for each individual eigenvector associated with the Size Function indexed by θ. 98

ϱ_θ Weighting factor for each set of eigenvectors associated with the Size Function indexed by θ. 98

ζ_θ Weighted Eigenspace size function calculated from size function $\ell_{\varphi\theta}$. 98

ζ Set of Weighted Eigenspace size functions calculated from the set of size functions Γ_φ. 105–108, 218, 220

Chapter 4: Spatiotemporal Gesture Recognition Glossary

Δ_y Set of Isolated samples of a sign class y. 126, 128, 130, 151–153, 156, 157, 159

Γ^T Temporal Clustering time scaling factor. 128

G Observation sequences $G = \{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_T\}$. 117, 119, 126–128, 134–139, 146, 147, 151, 156, 159, 168, 169, 174, 219, 220

M Dimension of observation vector $\mathbf{f}_t$. 117–119, 135

Ω Pre-defined threshold value for CRF likelihoods. 146, 153, 168

R State Reach: The number of states that it is possible to transition to from the current state. 126, 133

Σ_j Covariance matrix used to calculate probability distribution function for state s_j. 119, 125, 130

S Set of Hidden Markov Model States $S = \{s_1, s_2, ..., s_N\}$. 118, 126–128, 130, 133, 143, 150

b_j Observation probability distribution in state s_j. 118, 119

β_y Likelihood of gesture y relative to the movement epenthesis likelihood. 138, 139

$\mathbf{f}_t$ Spatiotemporal observation vector made at time t: $\mathbf{f}_t = \{o_1, ..., o_M\}$.

117–119, 128, 138, 153, 157, 160, 168, 219

κ_e Candidate Gesture End Point. 137, 138, 219

κ_p Candidate Gesture Likelihood. 139, 140, 168

κ_s Candidate Gesture Start Point. 137, 138, 219

κ Candidate Gesture. 137, 139, 168, 219, 220

Λ Set of Gesture HMMs and Threshold HMM $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_Y, \overline{\lambda}\}$. 134

$\overline{\lambda}$ Threshold Hidden Markov Model. 123, 133, 134, 136, 212

λ Hidden Markov Model. 118, 119, 127, 130, 133–137, 139, 212

μ_j Mean vector used to calculate probability distribution function for state s_j. 119, 125, 130

ω_{Ly} Variance weight for Left hand HMM. 135, 136, 146

ω_{Ry} Variance weight for Right hand HMM. 135, 136, 146

s Individual Hidden Markov Model state. 118, 123, 133

τ_y Movement Epenthesis likelihood threshold for gesture y. 134, 137, 139

Chapter 5: Weakly Supervised Training Glossary

B_i Bag representing a sequence of features for each frame $B_i = \{B_i[0], ..., B_i[N_{Bi}]\}$. 194, 197–199, 201

B^+ Set of positive bags $B^+ = \{B_0, ..., B_{N_B}\}$. 194, 198, 199, 223

$D^G()$ Similarity function used to calculate similarity between spatiotemporal gesture frames. 191, 195, 197

$D^H()$ Similarity function used to calculate similarity between hand posture frames. 192, 197

G_i Bag representing a sequence of spatiotemporal features for each frame $G_i = \{\mathbf{f}_i[0], ..., \mathbf{f}_i[N_{Bi}]\}$. 195–197, 199, 202, 205

G^+ Set of positive bags for spatiotemporal features$B^+ = \{B_0, ..., B_{N_B}\}$. 195

H_i Bag representing a sequence of hand posture features for each frame $H_i = \{C_i[0], ..., C_i[N_{Bi}]\}$. 195–197, 205, 213, 215, 219, 220

H^+ Set of positive bags for hand posture features$B^+ = \{B_0, ..., B_{N_B}\}$. 195, 213

$L(\xi_w)$ total number of hand postures in ξ_w. 213, 214, 216, 218

N_{Bi} Number of frames in bag B_i. 195, 196, 198, 199

N_B^w Number of bags in the set of bags B_w^+ where B_w^+ is the set of bags for target word w. 215

Ψ_i Overall density vector calculated from i'th hand posture and spatiotemporal density vectors. 202, 204, 213

Ψ Set of all density vectors $\Psi = \{\Psi_1, ..., \Psi_{N_B}\}$. 204

$\Psi(B_i)$ Function to compute the density vector for the i'th hand posture or spatiotemporal bag. 201, 213

S_w Measure of dissimilarity for cluster ξ_w. 214–216

Ξ Set of all key hand posture clusters $\Xi = \{\xi_0, ..., \xi_W\}$. 213, 216, 218

$\overline{B_{ij}}$ Comparision matrix used to compared bags B_i and B_j. 197, 198

$\overline{G_{ij}}$ Comparision matrix used to compared temporal bags G_i and G_j. 195, 197, 199, 202

$\overline{H_{ij}}$ Comparision matrix used to compared hand posture bags H_i and

H_j. 195–197

$\overline{\mathbf{b}_{ij}}'$ Transposed comparison vector used to compared bags B_i and B_j. 198

$\overline{\mathbf{b}_{ij}}$ Comparision vector used to compared bags B_i and B_j. 197, 198

$\mathbf{b}_j'$ N_{Bj}-dimensional column vector storing the summation of transposed comparison vectors $\overline{\mathbf{b}_{kj}}'$. 199

$\mathbf{b}_i$ N_{Bi}-dimensional row vector storing the summation of comparison vectors $\overline{\mathbf{b}_{ij}}$. 199, 201

$\Phi(B_i, B_j)$ $N_{Bi} \times N_{Bj}$ Density Matrix which measures the interaction between the pair of bags B_i and B_j. 201

$s_w[t]$ Measure of dissimilarity for t'th hand posture in cluster ξ_w. 214–216

t_i^{Max} Frame index of B_i with maximum density. 204

$\widehat{G_i}$ Subsequence of spatiotemporal gesture bag G_i which corresponds to the target word: $\widehat{G_i} = \{G_i[t_i^s], ...G_i[t_i^e]\}$ where t_i^s and t_i^e correspond to the start and end index of the target word. 205, 212, 213

$\widehat{H_i}$ Subsequence of hand posture bag H_i which corresponds to the target word: $\widehat{H_i} = \{H_i[t_i^s], ...H_i[t_i^e]\}$ where t_i^s and t_i^e correspond to the start and end index of the target word. 205, 212, 213

ξ_w Cluster of candidate key hand postures for word w. 213–216, 218

Abstract

This thesis presents a framework for the automatic recognition of Sign Language sentences. In previous sign language recognition works, the issues of; user independent recognition, movement epenthesis modeling and automatic or weakly supervised training have not been fully addressed in a single recognition framework. This work presents three main contributions in order to address these issues.

The first contribution is a technique for user independent hand posture recognition. We present a novel eigenspace Size Function feature which is implemented to perform user independent recognition of sign language hand postures.

The second contribution is a framework for the classification and spotting of spatiotemporal gestures which appear in sign language. We propose a Gesture Threshold Hidden Markov Model (GT-HMM) to classify gestures and to identify movement epenthesis without the need for explicit epenthesis training.

The third contribution is a framework to train the hand posture and spatiotemporal models using only the weak supervision of sign language videos and their corresponding text translations. This is achieved through our proposed Multiple Instance Learning Density Matrix algorithm which automatically extracts isolated signs from full sentences using the weak and noisy supervision of text translations. The automatically extracted isolated samples are then utilised to train our spatiotemporal gesture and hand posture classifiers.

The work we present in this thesis is an important and significant contribution to the area of natural sign language recognition as we propose a robust framework for training a recognition system without the need for manual labeling.

Chapter 1

Introduction

Sign languages are used all over the world as a primary means of communication by deaf people. For example, Irish Sign Language (ISL) is used by the majority of the hearing impaired community in Ireland. According to current estimates 1 in every 1000 people are deaf. In the United States alone, American sign language is used by more than 500,000 people on a regular basis with a further 1.5 million people using it from time to time. Thus, there is a great need for systems that can interpret sign language or can serve as interpreters between sign languages and spoken languages.

There is a limited number of hearing people who are competently able to communicate in sign language. Sign language interpreters can be used to aid communication between deaf and hearing people but this is often difficult due to the limited availability and high cost of interpreters. These difficulties in communication between hearing and deaf people can lead to problems in the integration of deaf people into society and conflicts with an

independent and self-determined lifestyle. Hearing people learn and perceive written language as a visual representation of spoken language where letters encode phonemes. For deaf people, this correspondence does not exist thus letters are just seen as symbols without any meaning [vAZC+08b]. Deaf people therefore have great difficulties with reading and writing due to the fact that there is no direct correspondence between their natural language (sign language) and written language. Research in automated recognition is therefore needed in order to improve communication between deaf and hearing people. Current developments in automatic sign language recognition is roughly 30 years behind speech recognition [vAZC+08b]. Sign language is conveyed through multiple interacting channels of information, therefore the analysis of sign language is a more complex problem than that of analysing the one-dimensional audio channel in speech.

The goals of the work discussed in this thesis is to develop:

1. Recognition models to process and classify the different channels of sign language communication.

2. Algorithms to train the recognition models with minimal human input.

This thesis describes three fundamental contributions to these goals:

1. The first contribution is the development of a framework for the automatic classification of hand shapes which are used in sign language communication. The framework developed is based on a novel eigenspace Size Function representation of the hand. The eigenspace Size Function acts as a user independent measurement of the hand and is implemented

to recognise the hand shapes of users not represented in the training set.

2. The second contribution is the development of a framework to segment and recognise motion based gestures from continuous sign language sentences. This framework is based on a Hidden Markov Model (HMM) framework where a Threshold HMM is developed to compute a dynamic threshold likelihood as a measure for inter-gesture transitions. This framework is implemented to spot and recognise meaningful gesture segments from within continuous sign language sentences.

3. The overall goal of sign language recognition requires the development of algorithms which scale to large vocabularies. There has been a significant amount of research in sign language recognition in recent years [OR05]. However, expanding sign vocabularies used in previous research has proven very difficult as previous works have typically required manual training data to be generated for each sign. The third contribution, and perhaps the most significant, is an approach towards automatic training of sign language recognition models. This thesis proposes a technique for the automatic extraction of sign training data from sign language videos with corresponding text translations. This technique is based on a novel Multiple Instance Learning (MIL) density matrix algorithm which automatically extracts isolated samples of signs that can then be used to train our hand shape and motion based gesture models.

1.1 Publications

Part of the work in this thesis has been presented in the publications listed in this section:

International Journal Publications

1. **D. Kelly**, J. Mc Donald and C. Markham,"A Person Independent System for Recognition of Hand Postures used in Sign Language", In Pattern Recognition Letters, Accepted

2. **D. Kelly**, J. Mc Donald and C. Markham,"Weakly Supervised Training of a Sign Language Recognition System using Multiple Instance Learning Density Matrices", In IEEE Transactions on Systems, Man and Cybernetics-Part B, In Submission

Book Chapter

1. **D. Kelly**, J. Mc Donald and C. Markham,"Recognition of Spatiotemporal Gestures in Sign Language using Gesture Threshold HMMs", In Machine Learning for Vision Based Motion Analysis, Springer LNCS, In Submission

International Conference Publications

1. **D. Kelly**, J. Reilly Delannoy, J. Mc Donald and C. Markham, "A Framework for Continuous Multimodal Sign Language Recognition", In Proceedings Special Interest Group on Computer-Human Interaction

2009, International Conference on Multimodal Interaction 2009, Boston MA.

2. **D. Kelly**, J. Mc Donald and C. Markham, "Evaluation of Threshold Model HMMs and Conditional Random Fields for Recognition of Spatiotemporal Gestures in Sign Language", In proceedings IEEE International Conference on Computer Vision 2009, Kyoto Japan.

3. **D. Kelly**, J. Mc Donald and C. Markham, "Continuous Recognition of Motion Based Gestures in Sign Language", In proceedings IEEE International Conference on Computer Vision 2009, Kyoto Japan.

4. **D. Kelly**, J. Reilly Delannoy, J. Mc Donald and C. Markham, "Incorporating Facial Features into a Multi-Channel Gesture Recognition System for the Interpretation of Irish Sign Language Sequences", In proceedings IEEE International Conference on Computer Vision 2009, Kyoto Japan.

5. **D. Kelly**, J. Reilly Delannoy, J. McDonald, and C. Markham,"Automatic Recognition of Head Movement Gestures in Sign Language Sentences", in CIICT 2009: Proceedings of the China-Ireland International Conference on Information and Communications 2009

6. **D. Kelly**, J. McDonald, and C. Markham, " Recognising Spatiotemporal Gestures and Movement Epenthesis in Sign Language," in IMVIP 2009: Proceedings of the Irish Machine Vision and Image Processing Conference 2009, 2009.

7. **D. Kelly**, J. McDonald, T. Lysaght, and C. Markham, "Analysis of sign language gestures using Size Functions and principal component analysis," in IMVIP 2008: Proceedings of the International Machine Vision and Image Processing Conference 2008, 2008.

8. **D. Kelly**, C. Markham, J. McDonald, "Demo Session - A System for Teaching Sign Language using Live Gesture Feedback", IEEE International Conference on Automatic Face and Gesture Recognition 2008, Amsterdam.

9. **D. Kelly**, J. McDonald, C. Markham, "A Hand Shape Classification Measurement Using Size Functions and Principal Component Analysis" Irish Graduate Student Symposium on Vision, Graphics and Visualisation 2008, Trinity College Dublin.

10. **D. Kelly**, P. Olivo, C. Markham, J Mc Donald, B. Caulfield and D. Fitzgerald, "Classification of Human Poses using a Vision Based Technique", in IMVIP 2007: Proceedings of the International Machine Vision and Image Processing Conference 2007, 2007.

11. **D. Kelly**, D. Fitzgerald, J. Foody, D. Kumar, T. Ward, B. Caulfield and C. Markham, "The E-Motion System: Motion Capture and Movement-based Biofeedback Game", in CGAMES 2006: Proceedings of the 9th International Conference on Computer Games, Dublin.

12. D. Fitzgerald, **D. Kelly**, T. Ward, C. Markham, B. Caulfield, "Usability Evaluation of E-Motion: A Virtual Rehabilitation System Designed to Demonstrate, Instruct, and Monitor a Therapeutic Exercise

Programme", in Proceedings Virtual Rehabilitation 2008, Vancouver, Canada.

13. D. Fitzgerald, J. Foody, **D. Kelly**, T. Ward, C. Markham, J. McDonald, B. Caulfield, "Development of a wearable motion capture suit and virtual reality biofeedback system for the instruction and analysis of sports rehabilitation exercises", in proceedings of the 29th Annual Conference of the IEEE Engineering in Medicine and Biology Society 2007, Lyon, France.

14. D. Fitzgerald, J. Foody, D. Kumar, **D. Kelly**, T. Ward, C. Markham, B. Caulfield, "Integration of kinematic Analysis into Computer Games for Exercise", in CGAMES 2006: Proceedings of the 9th International Conference on Computer Games, Dublin.

15. D. Kumar, J. Foody, D. Fitzgerald, **D. Kelly**, T. Ward, C. Markham, B. Caulfield, "Sensor Density Requirements for Kinematic Controllers in a Full Posture Yoga Gaming application", in CGAMES 2006: Proceedings of the 9th International Conference on Computer Games, Dublin.

16. J. Foody, **D. Kelly**, D. Kumar, D. Fitzgerald, B. Caulfield, T. Ward, C. Markham. "A real time motion capture system, using USB based tri-axis magnetic and inertial sensors, for movement based relaxation." Proceedings of the IET Irish Signals and Systems Conference, 2006, Dublin.

17. J. Foody, **D. Kelly**, D. Kumar, D. Fitzgerald, T. Ward, B. Caulfield,

C. Markham, "A Prototype sourceless kinematic feedback based video game for movement based exercise", in proceedings of the 28th Annual Conference of the IEEE Engineering in Medicine and Biology Society 2006, New York.

1.2 Outline of the thesis

The structure of the remainder of this thesis is as follows: Chapter 2 provides an overview of the linguistic properties of sign language, an analysis of the structure of sign language and a discussion of how particular signs are formed and distinguished from each other. We then present the major ideas in the state of the art in sign language recognition as well as a discussion of previously unsolved problems.

The development of a user independent hand posture recognition model is detailed in Chapter 3. A thorough evaluation of the discriminatory properties of our proposed hand posture features is discussed as well as an evaluation of our proposed hand posture recognition framework.

Chapter 4 details the implementation of our proposed motion based gesture spotter model. We implement a HMM based framework to spot and recognise meaningful gesture segments from within continuous sign language sentences. Moreover, an evaluation of our proposed model is carried out and compared to Conditional Random Field (CRF), Hidden Conditional Random Field (HCRF) and Latent Dynamic Conditional Random Field (LDCRF) frameworks.

In Chapter 5 we describe our technique developed for the automatic train-

ing of the sign language recognition models. Automatic training is performed using our MIL density matrix algorithm and a detailed explanation of the implementation of this algorithm is carried out. We discuss experiments conducted to evaluate the automatic training algorithm as well as experiments to evaluate the combination of our hand shape and motion recognition models when trained using the automatic training algorithm.

Chapter 6 concludes with a summary of our contributions and details of possible future directions of this work.

Chapter 2

Sign Language Recognition

2.1 Sign Language Overview

Gestures are a form of body language or non-verbal communication. Hand gestures can be classified into several categories such as conversational gestures, controlling gestures, manipulative gestures and communicative gestures [WH99].

Sign language is regarded as the most structured of all the gesture categories. Like spoken languages, sign languages emerge and evolve naturally within hearing-impaired communities. Wherever a hearing-impaired community exists, sign languages develop. Sign language develops independently from the spoken language of the region. Each sign language has its own grammar and rules, with the common property that they are all visually perceived. Like spoken language, there are many different sign languages of the world. For example, an Irish Sign Language signer could not understand

an American Sign Language signer unless they had specifically learned that language.

Although sign language is primarily communicated using hand gestures (manual signing), it also incorporates non-manual signals conveyed through facial expressions, head movements, body postures and torso movements. Due to the complexity and multimodal nature of sign language, the research area of Sign language recognition is a multidisciplinary research area involving pattern recognition, machine learning, computer vision, natural language processing and linguistics.

Sign language also have their own syntax and grammar. A misconception of sign language is that they are patterned after the vocally produced languages of that country, and that signs are manually manually produced English words. This is not the case and Sign Language have their own phonology, morphology, syntax and grammar that are independent of spoken languages. The morphological structure of sign language is simultaneous such that the different morphemes of a word are simultaneously superimposed on each other rather than being strung together, as those of spoken languages usually are. This is one of the main difference between signed and spoken languages. For example, manual signs are conveyed sequentially, where each sign comes one after the other. However, in addition to being conveyed sequentially, each manual sign occurs in parallel to manuals signs performed by the other hand as well as actions such as facial expressions or head and body movements. The linguistic characteristics of sign languages therefore differ greatly from those of spoken languages. Research has shown that this

morphological structure is not specific to any one sign language and therefore shows thats different sign languages have strong cross linguistic similarities in their morphological structures [AMS05].

In psycholinguistic research, there have been many studies on human gestures and on sign language in particular. One of the most important psycholinguistic works in sign language is the work Stokoe [Sto05]. In this work, Stokoe defined three aspects that are combined simultaneously in the formation of a particular manual sign: what acts, where it acts, and the act. These aspects translate into building blocks that linguists describe as: the hand shape, the position, the orientation and the movement. In sign language recognition these four manual sign components are often considered as two distinct information channels. The first channel is the hand posture channel, which refers to the finger configuration and orientation of the hand. The second channel is the spatiotemporal channel, which refers to the motion trajectory and location of articulation of the hands in space.

On their own, hand postures can be used for finger-spelling where different hand postures are used to represent the letters and numbers of writing and numeral systems. Finger-spelling can be used to convey words from a spoken language which have no sign equivalent, or for emphasis, clarification, or when teaching or learning a sign language.

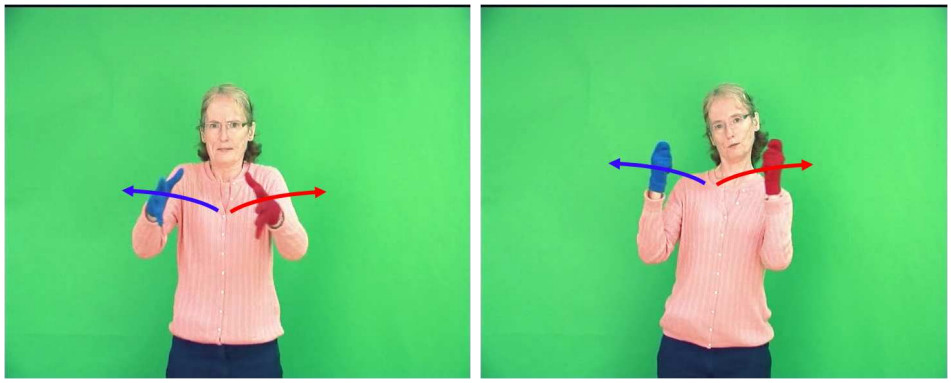
Sign language is a complex language and the majority of signs convey a significant amount of information in the combination of hand posture and hand motion. A large number of signs can only be distinguished when all the information from the manual channels are available. An example is illus-

trated in Figure 2.1, where the signs 'School' and 'Play' share the same hand postures but have different motions. Similarly, in Figure 2.2, the signs 'Big' and 'Paper' share the same motion and can only be distinguished by hand posture. In Figure 2.3, the signs 'Eat', 'Water', 'Warm' and 'Sweets' could only be distinguished by their hand shape. Therefore, recognising sign language communication requires simultaneous analysis of the spatiotemporal gestures and hand posture channels.

(a) Sign for 'School' (b) Sign for 'Play'

Figure 2.1: Examples of signs with similar hand posture. Signs must be distinguished using motion.

When spatiotemporal gestures are performed in a continuous sign language sentence, the hands need to move from the ending location of one sign to the starting location of the next. These inter-gesture transition periods are called movement epenthesis [SR89] and are not part of either of the gestures. Analysis of the spatiotemporal gesture channel must therefore distinguish between valid sign segments and movement epenthesis.



(a) Sign for 'Big' (b) Sign for 'Paper'

Figure 2.2: Examples of signs with similar motion. Signs must be distinguished using hand posture.

2.1.1 Applications of Sign Language Recognition

One of the main uses proposed for a sign language recognition system is a sign to text conversion system. This would require the complete translation of signed sentences to the text, or speech, of a spoken language. Such a translation system is not the only use for sign language recognition systems. There are other envisaged applications for sign language recognition systems such as a translation system for specific transactional domains such as post offices, banks etc. Another application is a bandwidth conserving system allowing communication between signers where recognised signs, which are the input of the communication system at one end, can be translated to avatar based animations at the other. An additional proposed application is an automated sign language teaching system. It could support users suffering from hearing loss, deaf people with sign language deficiencies and hearing

(a) Sign for ‘Eat’

(b) Sign for ‘Water’

(c) Sign for ‘Warm’

(d) Sign for ‘Sweets’

Figure 2.3: Examples of signs with similar location of articulation. Signs must be distinguished using hand posture.

people wishing to learn sign language.

Other envisaged applications include an automated, or semi-automated, system for the annotation of video databases of native signing. Linguistic research on sign language requires large scale annotated corpora and automated methods of analysing sign language videos would greatly improve annotation efficiency. Finally, sign language recognition systems could be incorporated into applications which enable an input interface for augmented communication systems. Assistive technology implemented for human to human communication by people with speech impairments often require keyboard, mouse and joystick inputs. Systems which could incorporate natural aspects of sign language would increase the accessibility of these systems.

The techniques proposed in this thesis are not limited to sign language recognition. Our proposed techniques have potential to be applied to different problems that focus on human motion modeling and recognition, such as gesture controlled Human Computer Interface (HCI) systems, human action analysis and social interaction analysis.

2.2 State of the Art on Sign Language Recognition

In this section we will review the state of the art literature on sign language and gesture recognition and highlight the issues within the current literature which we propose solutions to.

In order to build a framework for the automatic learning and recognition

of sign language, it is important that robust algorithms which model hand postures and spatiotemporal gestures be developed.

Research in the field of sign language recognition has made significant advances in recent years. Gesture recognition systems which deal with temporal gestures and hand postures are reviewed in this section. For a comprehensive review of automatic sign language recognition, refer to the survey paper by Ong and Ranganath [OR05].

2.2.1 Gesture Data Acquisition

The focus of the work detailed in this thesis is the development of computational models for the automatic learning and recognition of sign language information. In order to capture sign language information, data is mainly acquired using cameras or direct measure devices.

In this section we discuss some of the data acquisition methods using direct measure devices and cameras which have been implemented in the literature.

Wearable Computing Based Acquisition

Wearable computing approaches to sign language data acquisition offer accurate means of extracting information about the signers hand movements and hand shape.

Kim et al. [KWRA08] proposed a system which combined sensor data from accelerometers and an Electromyogram (EMG) which was used to measure the electrical activity produced by the hand muscles. It was shown that

the information added by the EMG greatly improved the recognition rate of signs. Figure 2.4 shows a visualisation of the sensor setup for a single hand.

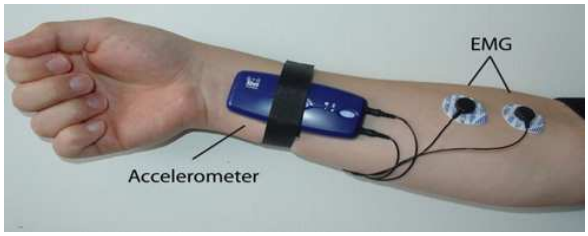


Figure 2.4: Sensor Placement for Bi-Channel recognition system proposed by Kim et al. [KWRA08]

Vogler et al. [VM04] recorded arm and hand movement data using "Ascension Technologies" MotionStar 3D tracking system and recorded hand posture information using a "Virtual Technologies" Cyberglove^{TM}. Fang et al. [FGZ03, GFZC04] developed a large vocabulary sign recognition systems using two Cybergloves^{TM} and three "Pohelmus" 3SPACE-position trackers. Two trackers were positioned on the wrist of each hand and another positioned on the signers back and were used to collect orientation and position data. The Cybergloves^{TM} collected 18-dimensional hand shape information for each hand. Similarly, Oz et al. [OL07] utilised a Cyberglove^{TM} along with a "Flock of Birds" 3-D motion tracker to extract hand posture features. Figure 2.5 shows the Cyberglove^{TM} and "Flock of Birds" 3-D motion tracker.

Another data glove based system is proposed by McGuire et al. [MHRS^{+}04] where a mobile sign language translator is implemented using an Acceleglove (See Figure 2.6). The Acceleglove consists of five micro two-axis accelerometers mounted on rings read finger flexion. Two more in the back of the palm

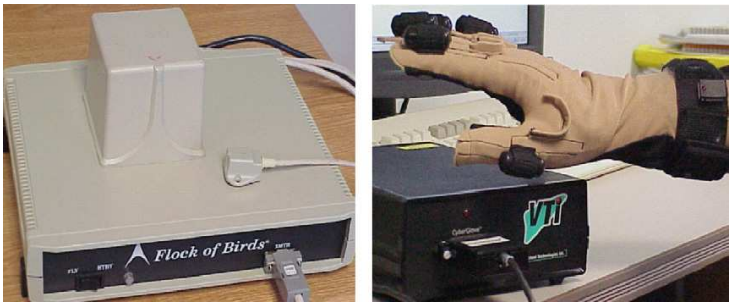


Figure 2.5: Left: Flock of Birds 3D motion tracker, Right: CyberGlove

measure orientation. Not shown in the figure are two potentiometers which measure bend at the shoulder and elbow and another two-axis accelerometer which measures the upper arm angles

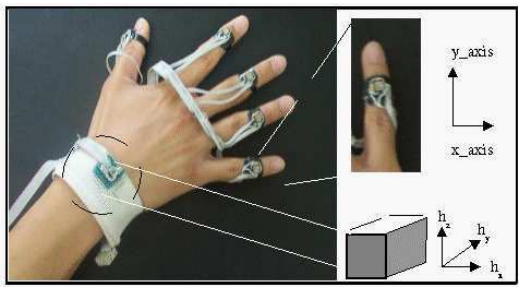


Figure 2.6: The Acceleglove

An novel approach to sign language data acquisition was taken by Brashear et al. [BSLJ03] where features from both a hat mounted camera and accelerometer data were used to classifying signs (see Figure 2.7).

Wang et al. [WCZ+07] proposed a viewpoint invariant data acquisition method. Their method was based on a virtual stereo vision system, using one camera and gloves with a specially designed colour pattern to indicate

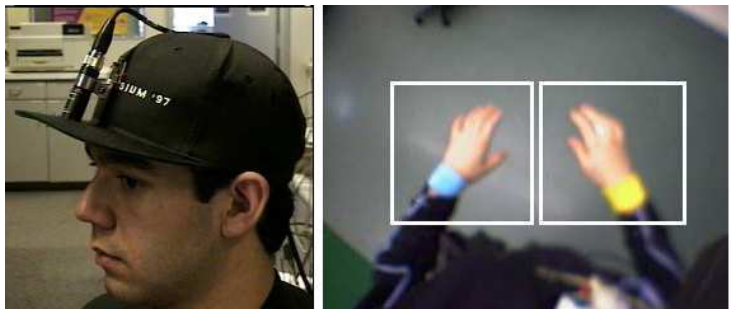


Figure 2.7: Head mounted camera and accelerometer data collection implemented by Brashear et al [BSLJ03]

the 5 separate fingers, palm and back. Figure 2.8 shows a visualisation of the design of the gloves.

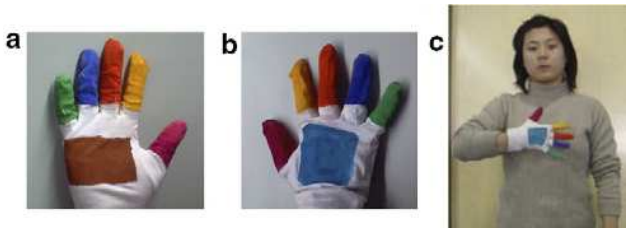


Figure 2.8: Gloves used by Wang et al. [WCZ+07]

Vision Based Acquisition

While wearable computing approaches to data acquisition can extract accurate features representing the signs being performed, some of these approaches require that the signer wears cumbersome devices which can hinder the ease and naturalness of signing. An alternative approach is to acquire gesture data through a camera based input. In order to capture gesture

based information from camera based inputs, the hands must be located in the image sequence and this is often carried out using colour, motion and edge information [OR05]. A number of works have proposed techniques for the segmentation of hands from an image sequence and in this section we will discuss some of these techniques:

Yang et al. [YSL09] implemented a skin colour and motion based segmentation method which included a displacement estimation used when the hands overlap the face. A template hand, stored on a previous frame, was used if the detected hand region was much larger than the hand region detected in the previous frame or the hand detector failed to detect the hand region.

Holden et al. [HLO05b] used a Principal Component Analysis (PCA) based skin colour model to identify the hands. Their method to segment occluded objects, using a combination of motion cues and the snakes algorithm, was used when hands and face overlap (see Figure 2.9).

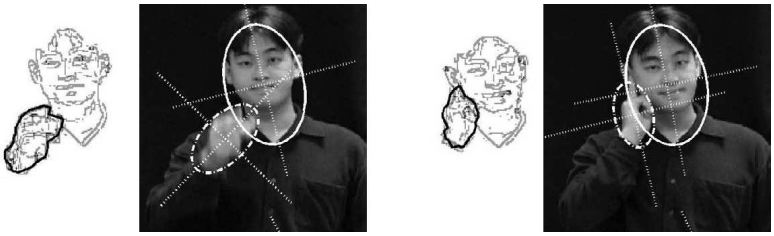


Figure 2.9: Examples of hand segmentation results from [HLO05b]

Cooper et al. [CB07] implemented a hand segmentation method using a skin colour model computed from the automatically detected face region. A model of the background was then created using a normalised histogram and

a threshold was applied to the likelihood ratio of face to background for each pixel (see Figure 2.10).

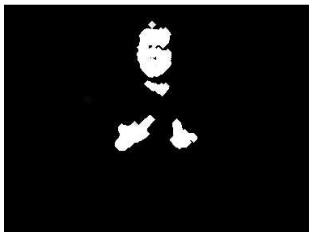


Figure 2.10: Examples of hand segmentation results from [CB07]

Askar et al. [AKE+04] proposed a skin colour segmentation method that automatically adjusted to the participant and illumination conditions. In order to account for skin region contact, such as overlapping head and hands, a set of rules were implemented to track the hand when hand and face contact occurred (see Figure 2.11).

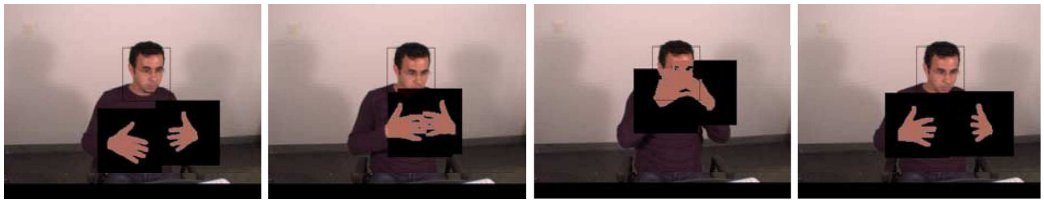


Figure 2.11: Examples of hand segmentation results from [AKE+04]

Barhate et al. [BPR+04] carried out hand segmentation using skin and motion cues within an on-line predictive Eigen-Tracking framework which approximated the hand motion by an affine transformation. Their method was shown to work well with occlusions and under poor illumination (see Figure 2.12).

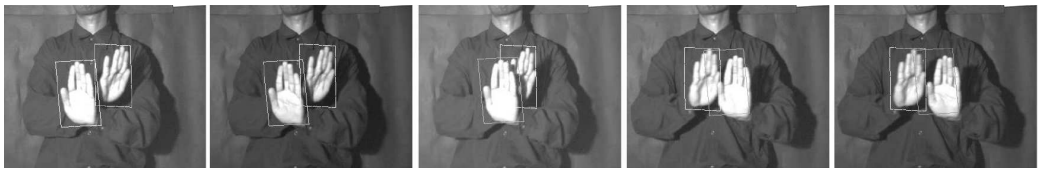


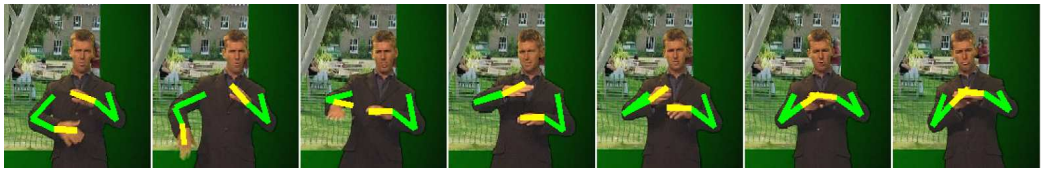
Figure 2.12: Examples of hand segmentation results from [BPR+04]

Donoser et al. [DB08] proposed a hand segmentation technique which combined skin colour likelihood maps with a modified version of the Maximally Stable Extremal Region (MSER) tracker. The MSER tracker found bright connected regions in the skin colour maps which had consequently darker values along their boundaries (see Figure 2.13).

Figure 2.13: Examples of hand segmentation results from [DB08]

Buehler et al. [BZE09] implemented an articulated upper-body model to track the head, torso, arms and hands of the signer. A graph cut method was used to segment the hand region predicted by the tracker into hand or background signer (see Figure 2.14).

Liwicki et al. [LE09] proposed a hand segmentation model where pixels



(a) Articulated Upper Body Tracking

(b) Graph Cut Segmentation

Figure 2.14: Examples of hand segmentation results from [BZE09]

were classified as hand or non-hand by a combination of three components: a signer-specific skin colour model, a spatially-varying non-skin colour model, and a spatial coherence prior (see Figure 2.15).

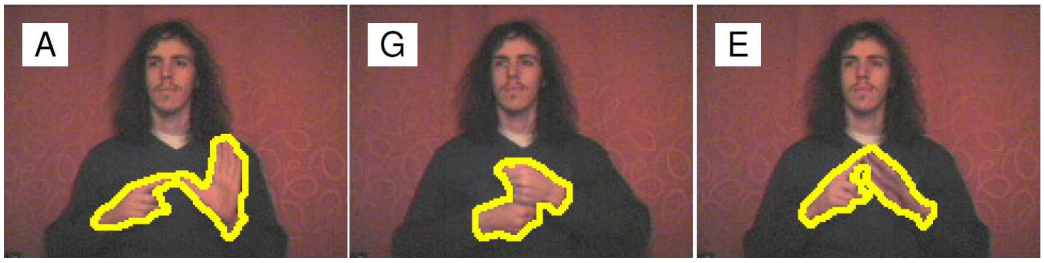


Figure 2.15: Examples of hand segmentation results from [LE09]

As we have discussed in this section, there are a variety of different techniques which have been proposed for the robust segmentation of hands from

image sequences. In order to achieve the full potential these segmentation methods have in the area of sign language recognition, we must develop algorithms which can recognise signs from the hand segmentation data. In this thesis we discuss our proposed set of techniques for the automatic learning and recognition of sign language. Our techniques are developed to utilise computer vision based hand segmentation data. We evaluate all of our proposed models using data extracted from image sequences, but the extraction techniques used are not the novel part of our work.

2.2.2 Spatiotemporal Gesture Recognition

The research into spatiotemporal gesture and sign recognition has two main categories: isolated and continuous recognition. Isolated recognition focuses on the classification of a single hand gesture. In continuous recognition, the user performs gestures one after the other and the aim is to spot and classify meaningful gesture segments from within the continuous stream of sign language.

Isolated Gesture Recognition

Yang et al. [YAT02] extracted motion trajectories from American Sign Language (ASL) videos and classified signs using a time delay neural network. Experiments based on a vocabulary of 40 signs showed the average recognition rate of unseen test trajectories was 93.4%.

Fang et al. [FGZ03] addressed the problem of large vocabulary sign recognition by proposing a combination of self organising feature maps, HMMs and

a hierarchical decision tree, with low computational costs, for the recognition of isolated signs. Experiments were conducted on a data set of 61365 isolated samples of 5113 different signs. Results showed an average recognition rate of 91.6%.

Juang et al. [JK05] proposed a recurrent fuzzy network for fuzzy temporal sequence processing. They applied their method to a gesture recognition task and experiments showed a recognition rate of 92%.

Agris et al. [vASZK06] proposed an isolated sign recognition system based on a combination of Maximum Likelihood Linear Regression and Maximum A Posteriori estimation. Their method was developed to consider the specifics of sign languages, such as one-handed signs. They implemented selected adaptation methods from speech recognition to improve the performance of their system when performing user independent recognition. A recognition rate of 78.6% was reported when recognising 153 isolated signs.

Shanableh et al. [SAAR07] proposed an isolated temporal gesture technique for the recognition of Arabic sign language. They proposed temporal features which were extracted through forward, backward, and bi-directional predictions. These prediction errors were thresholded and accumulated into one image that represented the motion of the sequence. Experiments, based on a database of isolated signs, showed that their method achieved a classification performance ranging from 97% to 100% when classifying 23 different sign classes.

Wang et al. [WCZ+07] proposed a view invariant sign recognition system. In their proposed system, the recognition task was converted to a verification

task based on the geometric constraint that the fundamental matrix associated with two views should be unique when the observation and template signs are obtained synchronously under virtual stereo vision and vise versa. Experiments conducted on a vocabulary of 100 signs, where 5 isolated samples of each sign were recorded, showed their method achieved an accuracy of 92%.

Cooper et al. [CB07] implemented an isolated sign recognition system using 1^{st} order Markov Chains. In their model, signs were broken down in visemes (equivalent to phonemes in speech) and a bank of Markov Chains were used to recognise the visemes as they were produced. Experiments, based on 5 unseen examples of each of the 164 signs in the vocabulary, showed a classification accuracy of 72.6%.

Kim et al. [KWRA08] evaluated an accelerometer and EMG based sign recognition system on 7 word level signs and results showed an average accuracy of 99.8% when tested on a total of 560 isolated samples.

Gunes et al. [GP09] propose an affect recognition system using hand gestures along with facial cues. Temporal segments of hand gestures and facial actions were detected using a HMM based system. Experiments showed that their proposed system achieved an accuracy of 88.5% when tested on isolated videos.

Ding et al. [DM09] develop a sign language recognition model which incorporated hand shape, motion and 3D position in a single framework. Signs were recognised using a tree based classifier where, for example, if two signs had a similar hand shape then the root of the tree would represent

the hand shape and the branches would represent different motion based gestures. A recognition rate of 93.9% was reported for a vocabulary of 38 signs.

While these works propose promising gesture recognition techniques, the experiments are based on isolated gesture samples. Natural gestures which occur in sign language are continuous. Therefore sign language recognition requires spotting of the gesture from continuous videos (i.e. determining the start and end points of a meaningful gesture pattern).

Continuous Gesture Recognition

Extending isolated recognition to continuous signing is difficult problem. It requires automatic detection of movement epenthesis segments so that the recognition algorithm can be applied to segmented signs.

One proposed solution to movement epenthesis detection is an explicit segmentation model were subsets of features from gesture data are used as cues for valid gesture start and end point detection. Oz et al. [OL07] proposed a continuous recognition framework which detected "signing" and "not signing" segments using a velocity network. The velocity network classified a "signing" segment from when the hand first showed a change in velocity until the time the velocity showed a series of low velocities. A Neural Network based classifier was trained to recognise 60 different one handed ASL signs. Experiments conducted on a total of 360 ASL words using histograms of feature vectors showed a recognition accuracy of 95%. The limitation of this explicit segmentation model arises from the difficulty in creating general

rules for sign boundary detection that could be applied to all types of manual and non-manual gestures [OR05]. For example, fluent signers perform sign language sentences in a very fluid and natural manner and sign boundaries often do not occur when there is a sharp change in hand velocity.

An approach to dealing with continuous recognition, without explicit segmentation, is to use HMMs for implicit sentence segmentation. Bauer et al. [BK02] and Starner et al. [SPW98] modeled each word or subunit with a HMM and then trained the HMMs with data collected from full sentences. Starner et al. [SPW98] conducted experiments on a vocabulary of 40 signs using a set of 478 sentences for training and testing. Results showed a word detection rate of 96.8%. A disadvantage of these methods is that training on full sentence data may result in a loss in valid sign recognition accuracy when tested on sentences not used in training. This is due to the large variations in the appearance of all the possible movement epenthesis that could occur between two signs.

Brashear et al. [BSLJ03] extended on the work of Starner et al. by developing a mobile sign recognition system. Their HMM based sign recognition system was implemented to recognise continuous sentences using camera and accelerometer inputs. Experiments, conducted on a vocabulary of 5 signs, showed a recognition accuracy of 90.5%. It was also shown that combining accelerometer and vision data improved the performance when compared to vision only data (52.4%) and accelerometer only data (65.9%).

Other works deal with movement epenthesis by explicitly modeling the movements between signs. Gao et al. [GFZC04] proposed a Transition Move-

ment Models (TMM) where transition HMMs were created to model the transitions between each unique pair of signs. The total number of TMMs were then reduced by a process of dynamically clustering the transition parts. An iterative segmentation algorithm was implemented to automatically segment the continuous sentences. Experiments, conducted on a set containing 3000 sentence samples with a vocabulary of 5113 signs from Chinese Sign Language (CSL), showed their method achieved an accuracy of 90.8%.

Vogler et al. [VM04] proposed a system to incorporate hand motion and hand posture data into a single recognition framework. A set of parallel HMMs were implemented to recognise signs from a vocabulary of 22 signs. Separate HMMs were implemented to model movement epenthesis between each unique ending and starting location of signs. Experiments showed their framework achieved a sign detection rate of 87.88% when tested on 99 sentences containing a total of 312 signs.

While these works, utilising explicit epenthesis models, have had promising results in gesture recognition and movement epenthesis detection, the training of such systems involves a large amount of extra data collection, manual data labeling and model training due to the extra number of HMMs required to detect movement epenthesis. Few researchers have addressed the problem of movement epenthesis without explicitly modeling these movements.

An novel approach to gesture/activity spotting was proposed by Junker et al. [JALT08] where a combination of explicit motion segmentation and HMM gesture classification was carried out. A pre-selection stage was implemented

in order to identify relevant motion events. These candidate motion segments were then classified in isolation using HMMs. Experiments conducted to evaluate the gesture spotting system showed that the method performed well when spotting gestures in 2 different activity scenarios. Results showed a total precision of 0.74 and a total recall of 0.93 for the first scenario and total precision of 0.73 and a total recall of 0.79 for the second scenario.

Another solution to segmenting signs from continuous streams of data, without modeling movement epenthesis, is to use grammar based information. Yang et al. [YSL07, YSL10] proposed an ASL recognition method based on an enhanced Level Building algorithm and a Trigram grammar model. Their method was based on a dynamic programming approach to spot signs without explicit movement epenthesis models. The recognition rate was 83% with 39 signs, articulated in 150 different sentences. Their work was based on a two step approach for the recognition of continuous signs, where the first step recognised the possible signs in the sentence and the second step applied a grammar model to the possible signs. They reported only the results obtained after the second step which applied a trigram grammar model to the signs. The reliance of the system to the grammar model was shown in the experiments where the recognition rate of the system decreased from 83% to 68% when the trigram model was replaced by a bigram model. Holden et al. [HLO05a] developed an Australian sign language recognition system where each sign is modeled using a HMM model. The recognition model employed grammar rules, based on 21 distinct signs, to recognise continuous sentences. Experiments showed their system achieved 97% recognition rate on 163 test

sign phrases, from 14 different sentences. It was noted in the work that the sign vocabulary used in experiments consisted of signs which where mainly distinguishable from motion alone.

Yang et al. [YSL09] proposed a very promising technique without the need for explicit epenthesis training or grammar rules. They develop threshold models in a CRF model which performed an adaptive threshold for distinguishing between signs in a vocabulary and non-sign patterns. Experiments showed their system could spot signs from continuous data with an 87.0% detection rate from a vocabulary of 48 signs where the system was trained on 10 isolated samples of each of the 48 signs. The system was then tested on continuous sentences which contained 480 samples of the signs in the sign vocabulary.

Non-Manual Signals

Recognising sign language communication requires simultaneous observation of manual and non-manual signals and their precise synchronisation and signal integration. Thus understanding sign language involves research in areas of face and facial expression recognition tracking and human motion analysis and gesture recognition.

Recently there has been a significant amount of research investigating the role of non-manual signals in sign language and attempting to quantify their individual importance. Works such as [Bah96, vdKCE06, BS86] focused on the role of head pose and body movement in sign language, where they reported a strong correlation linking head tilts and forwards move-

ments to questions or affirmations. The analysis of facial expressions for the interpretation of sign language has also received a significant amount of interest [GK06, GK07]. Computer-based approaches which model facial movement using Active Appearance Models (AAM) have been proposed [vAKK08, vAZC+08a, VG08]. Grossman et al. conducted an interesting study on ASL, where it was shown that eyebrow movement, and the degree of eye aperture movement, had a direct link to emotions and questions [GK06]. They showed that anger, wh-questions (who, where, what, when, why, how) and quizzical questions exhibited lowered brows and squinted eyes, while surprise and yes/no questions showed raised brows and widened eyes. The development of a system combining manual and non-manual signals is a non-trivial task [COR05]. This is demonstrated by the limited amount of work dealing with the recognition of multimodal communication channels in sign language. Ma et al. [MGW00] used HMMs to model multimodal information in sign language but lip motion was the only non-manual signal used. Their work was based on the assumption that the information portrayed by the lip movement directly coincided with that of the manual signs. While this is a valid assumption for mouthing, it cannot be generalised to other non-manual signals as they often span multiple manual signs and thus should be treated independently.

Issues Relevant to Spatiotemporal Gesture Recognition

The difficulty with recognising spatiotemporal gestures is that the hand must move from the end point of the previous gesture to the start point of the

next gesture. These inter-gesture transition periods are called movement epenthesis [SR89] and are not part of either of the signs. Thus, the issue with developing continuous recognition systems is creating algorithms which are be able to distinguish between valid sign segments and movement epenthesis.

As we have discussed, most previous work has required the explicit modeling of each epenthesis or required specific grammar rules. While these works have had promising results in gesture recognition and movement epenthesis detection, the training of explicit epenthesis models involved a large amount of extra data collection, manual data labeling, model training and recognition computation due to the extra number of HMMs required to detect movement epenthesis.

Another approach employed is to utilise grammar rules to reduce the number of possible sign combinations which appear in signed sentences. Grammar rules will become a more important aspect of sign language recognition when sign vocabularies grow to represent a large portion of the signs used in everyday sign language communication. State of the art sign recognition research is now at the stage where the main focus is on developing algorithms to model signs. It is difficult to evaluate sign recognition models which employ grammar rules on small sign vocabularies. For example, in a corpus of 30 signs which contains 8 nouns, if grammar rules are used to predict that the next sign is likely to be from the noun category, then the number of possible signs the recognition model must choose from is reduced to 8 signs. With the overall goal of large corpus sign recognition in mind, experiments should be conducted in order to evaluate recognition models in their ability

to distinguish one sign from as many other signs as possible. In the works discussed in Section 2.2.2, which employ specific grammar rules, it is unclear how these models would perform if the grammar models were created from a larger real world corpus.

Other works focus on explicit gesture segmentation to facilitate continuous recognition. Particular gesture cues, such as changes in hand velocity, are used to determine gesture start and end points. While these explicit segmentation methods have been shown to work well in general activity recognition tasks, creating explicit segmentation rules for the task of sign language recognition is impractical due to variability in gesture structure and speed which occurs in natural sign language.

Few researchers have addressed the problem of movement epenthesis without employing grammar or segmentation rules or explicitly modeling the epenthesis. We propose a solution to this by developing a spatiotemporal gesture model which addresses the problem of movement epenthesis detection. We develop a HMM based threshold model training and recognition framework to classify spatiotemporal gestures and to identify movement epenthesis without explicitly training on movement epenthesis examples. Our proposed models can effectively recognise gestures from within sign sentences independent of any grammar rules.

Furthermore, while non-manual signals are an important aspect of sign language recognition, very few works have accounted for these non-manual signals when developing sign recognition models. We also show that our spatiotemporal recognition model is scalable to modes of communication other

than manual signs by developing robust head movement and facial expression recognition models.

2.2.3 Hand Posture Recognition

In the previous section we reviewed techniques proposed for the recognition of spatiotemporal gestures. In this section we review literature aimed at developing techniques for the automatic recognition of hand postures.

To describe the shape of the hand, a number of methods for 2D shape representation and recognition are used. These include segmented hand images, binary hand silhouettes or hand blobs, and hand contours. Cui and Weng [CW00] used normalised segmented hand images as features and reported a 93.2% recognition rate on 28 different signs. Similarly Kadir et al. [KBOZ04] used normalised segmented hand images and greedy clustering techniques to recognise hand shapes with 75% accuracy. PCA has been shown to be successfully applied to gesture data to reduce dimensionality of the extracted features. Henrik et al. [HMM97] applied a normalising technique to segment hand images. PCA was then applied to extract features for recognition. User dependent tests on the system showed a 99% recognition rate on 25 hand gestures, while user independent tests reported a drop in recognition accuracy to 70%.

Deng and Tsui [DT02] applied a two-layer PCA / Multiple Discriminant Analysis scheme. A non user-independent experiment showed a recognition rate up to 70% on 110 signs. Imagawa et al. [IMT+00] calculated an eigenspace on segmented hand images and signs were then represented by

symbols which correspond to clusters. Results showed a recognition rate of 92% when tested on 33 signs. Patwardhan et al. [PR07] implemented a Predictive Eigen-Tracker to track the changing appearance of a moving hand. The algorithm obtained the affine transforms of the image frames and projected the image to the Eigenspace. An accuracy measurement of 100% was reported from tests using 80 gestures, although 64 of the test gestures were used in training and gestures used were very simple and distinct. Holden and Owens [HO03] presented a topological formation shape representation that measured the fingers only. A recognition rate of 96% was achieved when classifying 4 distinct hand shapes.

Contour based features have also been shown to perform well in hand posture recognition. Huang et al. [HH98] used Fourier descriptors of the hand contour, a Hausdorff distance measure and graph matching algorithms within a 3D Hopfield neural network to recognise signs with 91% accuracy. Al-Jarrah et al. [AJH01] extracted features by computing vectors between the contour's centre of mass and localised contour sequences. Recognition of 30 gestures was reported with an accuracy of 92.55%. Handouyahia et al. [HZW99] presented a sign language alphabet recognition system using a variation of Size Functions [UV95] called moment based Size Functions, which recognised 25 different signs with 90% accuracy.

Starner et al. [SPW98] showed that geometric moments perform well in hand gesture recognition. A head mounted camera tracked the signer's hands using skin colour. Hand blobs were extracted from video sequences and a 16 element geometric moment feature vector was used to describe the hand

shape. A recognition rate of 98% for sign language sentences was reported. Tanibata et al. [TSS02] used a set of 6 geometric moments to recognise Japanese sign language, although it was reported that recognition performed well, no recognition accuracy was specified. Bauer and Heinz [BH00] described a German sign language recognition system where hand shape feature experiments showed that the area of the hands performed well as a hand shape feature. It was reported that the system achieved an accuracy of 75% when only taking hand area into account. All training and test data was recorded from the same subject performing signs and a recognition rate of 94% and 91.7% was reported for systems based on a 52 and 97 sign lexicon respectively. Liwicki et al. [LE09] proposed a system to classify finger spelled words based on different visual features and a HMM framework. Signer-dependent experiments showed a recognition rate of 98.9% when tested on a lexicon of 100 signs.

All the hand posture recognition techniques we have discussed thus far have evaluated their methods on training and test sets recorded by the same people. Analogous to speaker independence in speech recognition, an ideal sign recognition system should give good recognition accuracy for signers not represented in the training data set. User independent hand posture recognition is particularly challenging as a user independent system must cope with geometric distortions due to different hand anatomy or different performance of gestures by different people.

Farhadi et al. [FFW07] proposed a signer-independent ASL transfer learning model to build sign models that transfer between signers. Results

showed their method achieved a classification accuracy of 67% when classifying signs from a 90 word vocabulary, but their method does not explicitly deal with hand posture recognition. Licsár et al. [LS05] developed a hand gesture recognition system with interactive training. Their proposed solution to user independent hand posture recognition is based on the concept of an on-line training method embedded into the recognition process. The on-line training is interactively controlled by the user and it adapts to his/her gestures based on user supervised feedback where the user specifies if detected gesture were incorrectly classified. This method was shown to work very well in the scenario where the hand posture recognition system is being used as a HCI interface for a camera-projector system allowing users to directly manipulate projected objects with the performed hand gestures. While it is feasible to implement online retraining of gestures based on supervised user feedback in this HCI scenario, implementing this model in an automatic sign language recognition system would make the performance of sign language un-natural and thus is not an appropriate option for this work.

Triesch et al. [TvdM02] proposed a user independent hand posture recognition system using Elastic Graph Matching which reported a recognition rate of 92.9% when classifying 10 hand postures performed by 24 different subjects. Just et al. [JRM06] recognised the same set of hand postures used by Triesh et al. using the Modified Census Transform with a recognition rate of 89.9%.

Flasinski et al. [FM10] developed a novel technique to generate structured graph descriptions of the hand. The graph descriptions were then analysed

using ETPL(k) graph grammar parsing. User independent experiments conducted on a set of 720 images, comprising of 10 different Polish sign language hand postures, showed an average recognition rate of 94%.

Issues Relevant to Robust Hand Posture Classification

The second problem with developing automated techniques for sign recognition is that an ideal sign recognition system should give good recognition accuracy for signers not represented in the training data set. User independent hand posture recognition is particularly challenging as a user independent system must cope with geometric distortions due to different hand anatomy or different performance of gestures by different people. Not only do hand postures express some concepts but they can also act as special transition states in temporal gestures. Therefore, finding techniques to combine these hand posture and temporal gesture channels is an important and difficult challenge in the automatic recognition of sign language.

As we have discussed, the majority of the literature on hand posture recognition has presented performance measures which where results of signer-dependent experiments carried out by testing the system on subjects who were also used to train the system. Of the few user independent hand posture recognition systems, the elastic graph matching method, proposed by Triesch et al. [TvdM02], showed very promising results. The disadvantage of the elastic graph matching algorithm is that it was reported to have a high computational complexity with the method requiring several seconds to analyse a single image.

We propose a solution to these problems by developing a novel hand posture feature, an eigenspace Size Function. We develop an accurate user independent hand posture recognition system, utilising the eigenspace Size Function, which can classify hand postures in real time allowing the classification of hand images from continuous video streams.

2.2.4 System Training

Previous research on the recognition of temporal gestures and hand postures, described in Sections 2.2.2 and 2.2.3 carry out training using manually labeled training data.

An approach to reduce the number of manually labeled training samples needed to train these systems is to generate synthetic gesture samples. Jiang et al. [JGY+09] proposed a synthetic sign generation technique where small numbers of samples for each sign were collected using standard manually labeled data. Using a mean shift based outward generation algorithm, new synthetic sign samples, unspecific to the signer, were generated and used to train an isolated HMM based sign recognition system. Experiments showed that the addition of the synthesised data improved recognition accuracy.

Automatically labeling sign data, without the need for initial manually labeled data, is an extremely challenging task and is demonstrated by the limited works dealing with this problem.

Farhadi et al. [FF06] proposed a technique to align signs with English subtitles. A HMM based system was implemented using static and dynamic features. The HMMs were used to find the start and end of a sign and

a discriminative word model was built to perform word spotting. In their experiments, word spotting was carried out over an 80000 frame film.

Buehler et al. [BZE09] developed a weakly supervised technique, using MIL, to label start and end points of target sign language words from videos annotated with weakly aligned subtitles. This technique allowed the automatic extraction of isolated signs without manual labeling. Results showed that their technique was able to find 65% of the words from a vocabulary of 210 words.

Cooper et al. [CB09] also implemented an automated method to label start and end points of signs from videos using subtitles. A temporally constrained adaptation of apriori mining was used to extract similar regions of video, with the aid of a proposed contextual negative selection method. The system was tested on 23 signs which occurred in a 30 minute video. Their proposed method was able to correctly isolate, on average, 53.7% of the signs.

Nayak et al. [NSL09] proposed an unsupervised approach to extract and learn models for continuous basic units of signs, which are called signemes, from continuous sentences. They automatically extracted a signeme model, using Iterative Conditional Models, given a set of sentences with one common sign. Experiments showed their method was able to correctly extract 10 key signs from 136 sentences with an accuracy of 87%.

Issues Relevant to Weakly Supervised Training

The third problem in developing automatic sign language recognition systems is developing algorithms which scale to large vocabularies. A difficulty with this is that previous research has typically required manual training data to be generated for each sign. All works detailed in Sections 2.2.2 and 2.2.3 involved manual labeling and training where a signer, or interpreter, hand labeled each sign, or sign phoneme, such that a recognition system could be trained on isolated samples of each sign. This can be a very time consuming and expensive procedure and makes it difficult to expand sign vocabularies. Generating synthetic data can reduce the number of isolated samples needed but these methods still require initial sign samples to be labeled.

A solution to automatically acquiring training samples is to extract signs from sign language videos by utilising text translations associated with the sign language video. Automatic labeling of signs is an extremely challenging task and is demonstrated by the limited works dealing with this problem. While there exists a small number of works, discussed in this Section, which automatically label signs using text translations, no work has further developed these techniques to automatically train a full sign language spotting system on spatiotemporal and hand posture information. This thesis presents the first automatic sign labeling framework used to train a full sign recognition system on spatiotemporal and hand posture data. No other works exist which investigate the use of automatically labeled data in training natural sign language recognition systems.

We address the problem of automatic training by proposing a weakly

supervised system, using our proposed novel MIL density matrix algorithm, which automatically extracts isolated samples of signs that can then be used to train our gesture models. We utilise this training model, along with our proposed hand posture and spatiotemporal gesture models, to automatically train and classify natural sign language sentences.

2.3 Conclusion

The problem of sign language training and recognition can be defined as the analysis of all components that form the language and the understanding of individual signs or whole sequences of sign language communication. The overall aim in sign language recognition is to reach a large-vocabulary recognition system which would ease the communication of hearing impaired people with other people or with computers.

Ultimately, an ideal sign language recognition system is one that takes as input a sign language video and outputs a text interpretation of the sign language sentences. In order to achieve this goal, there are a number of issues which have not been fully addressed in the literature discussed in Section 2.2. In this chapter we have discussed the state of the art in sign language recognition and identified issues, which if solved, would make an important contribution to the area of sign language recognition. In this thesis we propose solutions to the problems of: continuous sign language modeling, robust hand posture classification and weakly supervised training.

In Chapters 3 and 4 of this thesis we will first present our hand posture and spatiotemporal recognition models before discussing our proposed

framework for the automatic training of the models in Chapter 5.

Chapter 3

Hand Posture Recognition [1]

3.1 Introduction

Hand gestures can be classified as either hand postures (hand shape and orientation) or temporal gestures (movement and position)[WHM99]. Hand postures not only express some concepts, but also act as special transition states in temporal gestures. Therefore, recognising hand postures is an integral requirement in gesture recognition.

One problem which we have highlighted as an issue relevant to developing robust sign recognition systems is that an ideal sign recognition system should give good recognition accuracy for signers not represented in the training data set. User independent hand posture recognition is particularly challenging as a user independent system must cope with geometric distor-

[1]The completed work discussed in this chapter has been accepted for journal publication: **D. Kelly**, J. Mc Donald and C. Markham,"A Person Independent System for Recognition of Hand Postures used in Sign Language", In Pattern Recognition Letters, Accepted

tions due to different hand anatomy or different performance of gestures by different people. In Chapter 2 we have discussed that there is a limited number of works which have developed hand posture recognition techniques that are capable of performing user independent hand posture recognition. In this chapter, we propose a solution to this problem by developing a novel hand posture feature, an eigenspace Size Function, which is robust at classifying hand postures independent of the person performing them.

3.2 Hand Features

In this chapter we propose a shape representation and pattern recognition framework to classify segmented hand images. In Section 2.2.1 we discussed a variety of different hand segmentation techniques which were proposed in the literature. These techniques can be utilised to extract binary images of a users hand from a video stream. In order to utilise the hand segmentation methods to their full potential, we develop a technique which can accurately classify hand postures from the hand segmentation data. In this chapter, it will be shown how our proposed hand shape representations are computed from a segmented hand contour. Our proposed hand shape features are developed to utilise computer vision based hand segmentation data. We evaluate our techniques using data extracted from image sequences, but it is the hand features that are the novel aspect of this chapter and not the extraction techniques used to evaluate the features.

In this chapter, we show how our proposed hand shape features are computed from a segmented hand contour. A thorough evaluation of the dis-

criminatory properties of our proposed features is carried out followed by an evaluation of our proposed hand posture recognition framework.

3.3 Shape Representations

Appearance based gesture recognition requires an accurate extraction of an effective feature set that can separate the hand shapes [PSH97]. We present a method of hand shape representation computed from the raw binary image and external contour extracted from the image. We propose a novel eigenspace Size Function shape representation which is calculated from the external contour. A Hu Moment feature set is also generated from the raw binary image.

Accurate shape representations must be able to identify similar shapes and distinguish between different shapes, therefore performance tests on different variations of our proposed shape representation will be carried out with the goal of achieving the optimal hand shape representation.

3.3.1 Review of Shape Representation Techniques

There have been many shape representation and description techniques proposed for many different visual feature recognition applications [ZL04]. Various shape techniques have been proposed, including shape signature, signature histogram, shape invariants, moments, curvature, shape context, shape matrix, spectral features etc. In general, shape representation techniques can be classified into two class of methods: contour-based and region based meth-

ods. Under each class, the different methods are further divided into structural approaches and global approaches. This sub-class is based on whether the shape is represented as a whole or represented by segments/sections.

Global Contour Descriptors

Some common simple global contour descriptors include: area, circularity, eccentricity, major axis orientation and bending energy. These simple global descriptors usually can only discriminate shapes with large differences and therefore are not suitable as stand-alone shape descriptors.

Another global contour descriptor is a technique known as correspondence-based shape matching which measures similarity between shapes using point-to-point matching. Hausdorff distance is type of correspondence-based shape matching and has been used to locate objects in a image and measure similarity between shapes [HKR93, CK99, Ruc97].Shape matching using Hausdorff distance is sensitive to noise and slight variations in shape.

Shape signatures represent another class of global contour descriptors. Shape signatures are calculated from a one dimensional function derived from the shape boundary points. Many shape signatures exist, they include centroidal profile, complex coordinates, centroid distance, tangent angle, cumulative angle, curvature, area and chord-length [ZL02]. shape signatures are sensitive to noise, and slight changes in the boundary can cause large errors in matching. Therefore, it is undesirable to directly describe shape using a shape signature. Further processing is necessary to increase its robustness and reduce the matching load.

Spectral descriptors overcome the problem of noise sensitivity and boundary variations by analyzing shape in spectral domain. Spectral descriptors include Fourier descriptor (FD) and wavelet descriptor (WD), they are derived from spectral transforms on 1-D shape signatures.

Elastic matching, a more robust global contour descriptor, was proposed by Bimbo and Pala [DBP97]. In this approach, a deformed template is generated as the sum of the original template and a warping deformation. As discussed in Section 2.2.3, Triesch et al. [TvdM02] have shown that this method works well for the application of hand posture recognition.

Global Region-based Descriptors

In region based techniques, all the pixels within a shape region are taken into account to obtain the shape representation, rather than only use boundary information as in contour base methods.

Geometric moments have been widely used for a number of different shape analysis applications [LP96]. Using nonlinear combinations of lower order moments, a set of moment invariants which has desirable properties of being invariant under translation, scaling and rotation, are derived. The main problem with geometric moments is that the few invariants derived from lower order moments are not sufficient to accurately describe shape on their own.

Orthogonal moments, an alternative to geometric moments, was proposed by Teague [Tea80]. Teague extended the idea of algebraic moments to a more general form and introduced Legendre moments and Zernike moments. Or-

thogonal moments allow for accurate reconstruction of the described shape, and makes optimal utilization of shape information. Although Zernike moment descriptor has a robust performance, it has several shortcomings. First, the kernel of Zernike moments is complex to compute, and the shape has to be normalized into a unit disk before deriving the moment features. Second, the radial features and circular features captured by Zernike moments are not consistent, one is in spatial domain and the other is in spectral domain. It does not allow multi-resolution analysis of a shape in radial direction. Third, the circular spectral features are not captured evenly at each order, this can result in loss of signi1cant features which are useful for shape description.

In this work we use a combination of a global contour based descriptors (size functions) and a global region based descriptors (Hu-moments). In the remainder of this Section we discuss these techniques and how they applied to hand posture recognition.

3.3.2 Hu Moments

Hu Moments [Hu62], which are a reformulation of the non-orthogonal centralised moments, are a set of transition, scale and rotation invariant moments. The set of Hu Moments, $I = \{I_1, I_2, I_3, I_4, I_5, I_6, I_7\}$, are calculated from the hand contour.

3.3.3 Size Functions

Size Functions are a technique used for shape description. They are integer valued functions which represent both qualitative and quantitative properties

of a visual shape by counting certain connected components of a topological space [VUFF93].

For a given contour C, extracted from the binary image of a hand, let G be a graph whose vertices are the points of the contour. Let φ, the measuring function, be a real-valued function defined on the vertices of G (see Figure 3.1(a)). The Size Function ℓ_φ induced by the measuring function φ, is an integer valued function defined on a real pair (x, y) according to the following algorithm:

1. Find the subgraph $G_{\varphi \leq y}$ of G determined by the points p with $\varphi(p) \leq y$ (see Figure 3.1(b));

2. Identify the connected components of $G_{\varphi \leq y}$ (see Figure 3.1(b));

3. The Size Function ℓ_φ at the point (x, y) equals the number of connected components of $G_{\varphi \leq y}$ which contain at least a vertex with $G_{\varphi \leq x}$ (see Figure 3.1(c), 3.1(d) and 3.1(e));

When identifying the number of connected components of the graphs $G_{\varphi \leq y}$ and $G_{\varphi \leq x}$ it should be noted that the graphs are circular. Therefore, in Figure 3.1(d), there exists 3 connected components of $G_{\varphi \leq y}$ which contain at least a vertex with $G_{\varphi \leq x}$, and not 4 which would be the case if the graphs where not circular. This ensures that the number of connected components will remain the same independent of the start and end point for which the measuring function was computed.

The theory of Size Functions does not identify a formal tool to resolve a suitable measuring function. Therefore, a suitable measuring function must

be found heuristically. As defined by Stokoe's model [Sto05], a hand posture is made up of the shape and orientation of the hand. Thus, for the application of classifying hand postures performed in sign language, the measuring function chosen must be sensitive to orientation changes of the hand. However a suitable classifier should not be sensitive to minor changes in hand orientation. With Stokoe's model in mind, the measuring function model proposed in this work utilises a family of measuring functions indexed by the angle $\theta\epsilon\{0, 1\frac{2\pi}{N_\Theta}, 2\frac{\pi}{2N_\Theta}, ..., (N_\Theta - 1)\frac{2\pi}{N_\Theta}\}$, where N_Θ is the total number of rotation angles used. Each measuring function $\varphi_\theta(p)$ is a function which rotates p about the centre of gravity of G and measures the distance between the horizontal axis and a point p on the rotated graph G_θ. The horizontal axis is a line which passes through the minimum point of G_θ. For every θ, a Size Function $\ell_{\varphi\theta}$ is generated, resulting in a set of Size Functions $\Gamma_\varphi = \{\ell_{\varphi 1}, \ell_{\varphi 2}, ..., \ell_{\varphi N_\Theta}\}$. The sensitivity of the system to orientation can then be controlled by means of adjusting N_Θ. As N_Θ increases, the number of rotations and Size Functions grows and the margin between each θ decreases. As the margin between each θ decreases, the effect small changes in orientation have on the final classification increases.

To illustrate the concept of Size Functions and their application in analysing hand postures utilised in sign language, a specific example will be used. For this example, let $N_\Theta = 4$. The hand contour is rotated to each of the four θ values (see Figure 3.2), the measuring function is applied to each of the four rotated contours (see Figure 3.3) and the Size Functions are then generated from each of the measuring functions (see Figure 3.4).

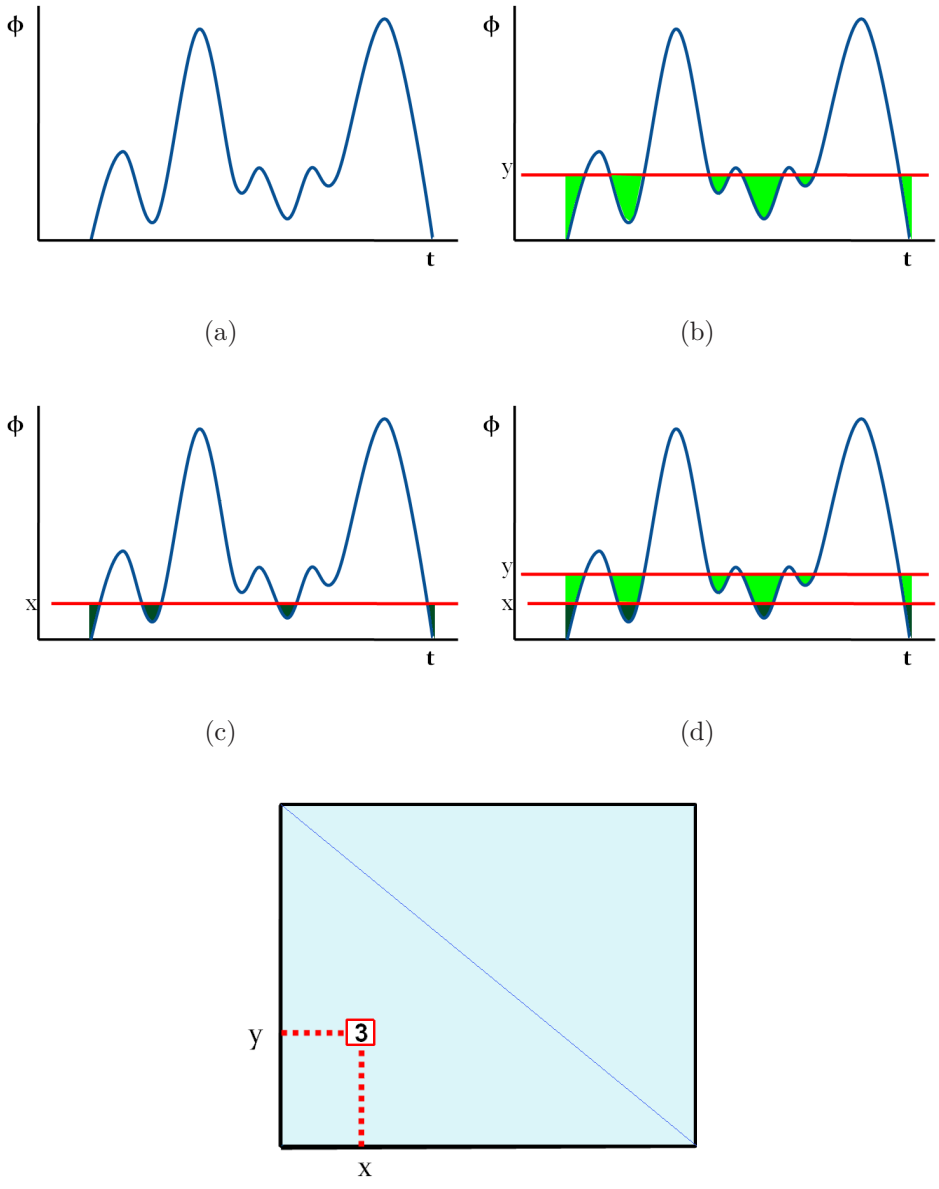


Figure 3.1: (a)Graph of some measuring function φ (b)Shaded region $\equiv \varphi \leq y$. (c)Shaded region $\equiv \varphi \leq x$. (d)Graph depicting $\varphi \leq y$ and $\varphi \leq x$. (e)Graph of Size Function l_φ with current $l_\varphi(x, y) = 3$.

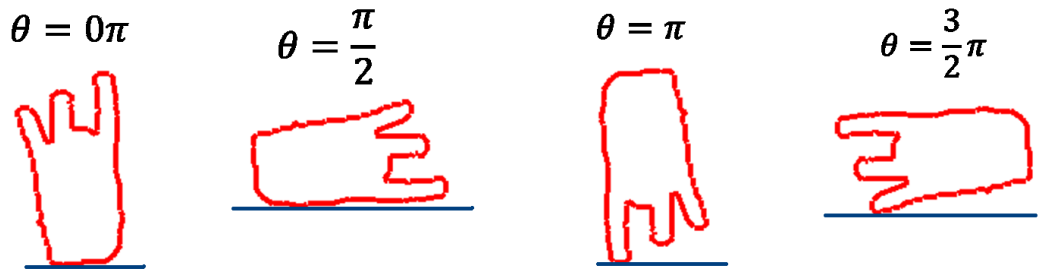


Figure 3.2: θ rotation applied to hand contour

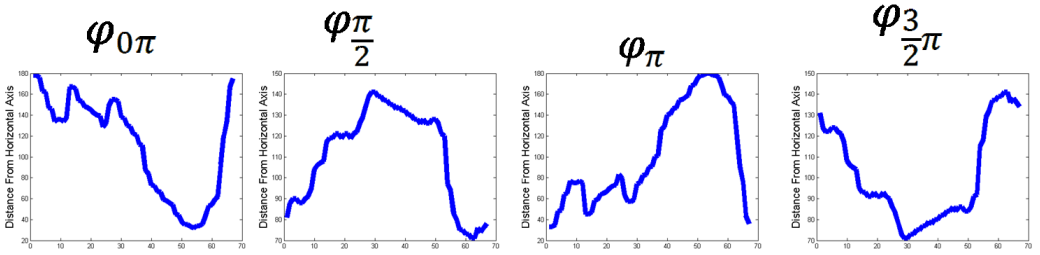


Figure 3.3: Measuring function φ_θ applied to hand contour

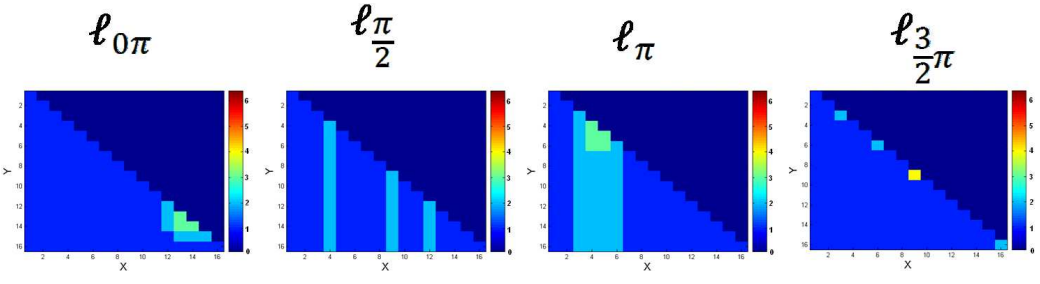


Figure 3.4: Size Function $\ell_{\varphi\theta}$ generated

3.3.4 Eigenspace Size Functions

In order to quantify the shape information held in a Size Function, we propose a more robust method of shape representation, as compared to the unmodified normalised Size Function representation used in [VUFF93][HZW99]. We make an important improvement to the Size Function technique by developing a Size Function feature which is more robust to noise and small changes in shape due to interpersonal differences. Our technique is a method of incorporating eigenspace information into the hand posture feature using PCA. PCA computes a linear projection from a high dimension input space to a low dimensional feature space. It is a statistical technique used for finding patterns in data of high dimensions. Since we are looking for similarities and differences between two Size Functions, we can utilise PCA in order to reduce the influence of noise, and small variations in shape by different people, and highlight portions of the Size Function useful for user independent hand posture recognition.

To calculate the principal components of a Size Function, the Size Function is described as an $N \times N$ matrix $X_\theta = \ell_{\varphi\theta}$. The vector $\boldsymbol{u}$ is the empirical mean of X_θ (see Equation 3.1), B_θ is the mean subtracted $N \times N$ matrix (see Equation 3.2) and Σ_θ is the covariance matrix of B_θ (see Equation 3.3).

$$\boldsymbol{u_\theta}[m] = \frac{1}{N} \sum_{n=1}^{N} X_\theta[m, n] \tag{3.1}$$

Where m and n refers to the row index and column index of the $N \times N$ matrix respectively and N refers to the width and height of the Size Function.

$$B_\theta = X_\theta - [\boldsymbol{u_\theta}, \boldsymbol{u_\theta}, ..., \boldsymbol{u_\theta}] \tag{3.2}$$

$$\Sigma_\theta = \frac{1}{N} B_\theta \cdot B_\theta^T \tag{3.3}$$

The eigenvectors and eigenvalues of Σ_θ are calculated according to Equation 3.4 were $\boldsymbol{v}$ is the eigenvector and w is the eigenvalue associated with the eigenvector.

$$\Sigma_\theta \boldsymbol{v} = w\boldsymbol{v} \tag{3.4}$$

The columns of the eigenvector matrix $M_\theta = \{\boldsymbol{v_1}, ..., \boldsymbol{v_N}\}$ and the eigenvalue matrix $W_\theta = \{\boldsymbol{w_1}, ..., \boldsymbol{w_N}\}$ are sorted in order of decreasing eigenvalue. This records the components in order of significance, the eigenvector with the highest eigenvalue being the *principal component*. Therefore the first column of M_θ, a $1 \times N$ vector, corresponds to the principal component vector.

Figure 3.5 shows a Size Function which was reconstructed with varying numbers of components. It should be noted that the reconstructed Size Functions are not used as features, we show the reconstructed Size Functions in order to illustrate the effect PCA dimensionality reduction has on the Size Function. The eigenvectors used to reconstruct the Size Functions are the features we use to recognise hand shapes.

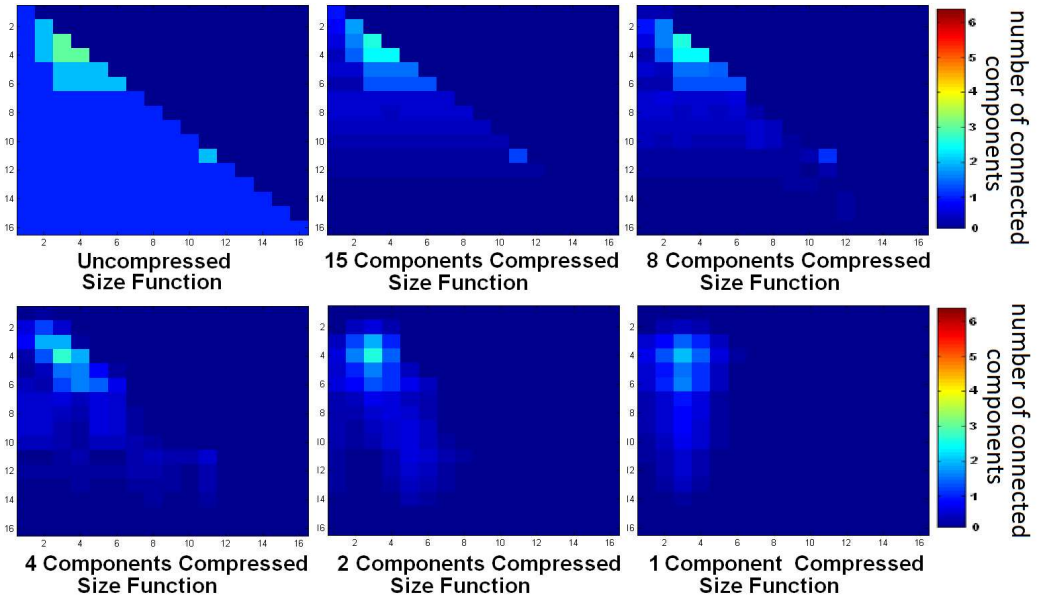


Figure 3.5: Size Functions Reconstructed with varying numbers of Components

3.4 Data Sets for Experimentation

In order to robustly evaluate our proposed hand posture classification techniques, we test our techniques using hand shape videos and images from two separate data sets. In this section we describe the two data sets.

3.4.1 Jochen Triesch Static Hand Posture database

The first data set is a benchmark database called the Jochen Triesch Static Hand Posture database [TvdM02]. We utilise this data set in order to evaluate our hand posture recognition framework and directly compare our system to other hand postures recognition research. The database consists of

10 hand signs (see Figure 3.6) performed by 24 different subjects against different backgrounds. All images are greyscale images and the backgrounds are of three types: uniform light, uniform dark and complex. In our system, posture recognition is carried out independent of hand segmentation. Neither motion nor colour are available in this data set, but, in general, colour and motion are two important cues needed to segment the hands from complex backgrounds and this is acknowledged by Triesch et al. [TvdM02]. Since there is no motion or colour cues available, we do not consider the hand images with complex backgrounds. It is still possible to make a like with like comparison with other research in this area as most results in the literature report recognition rates achieved on the uniform background images independent of complex background images. In this data set, we extract contours from each image by segmenting the image using Canny edge detection [Can86] and extracting the contour from the edge detected image using a border following algorithm [Sb85] (see Figure 3.7).

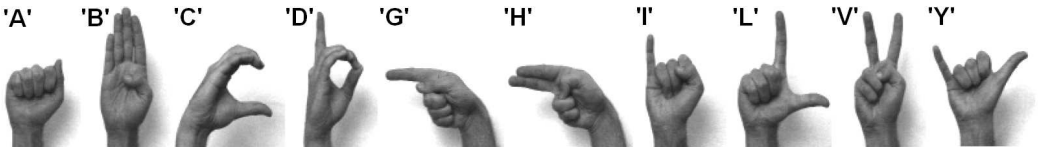


Figure 3.6: The ten postures of the Triesch data set, performed by one subject against uniform light background

3.4.2 ISL data set

The second data set is an in house ISL data set which we recorded in order to further evaluate our hand posture classification features. The data set

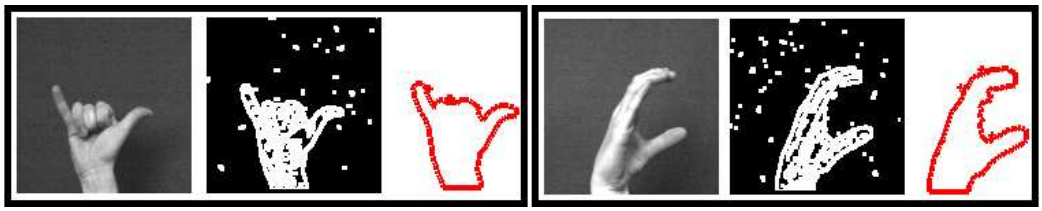


Figure 3.7: Example of Contour Extraction from 'Y' and 'C' hand postures from Triesch data set

consists of 23 hand signs (see Figure 3.8), from the ISL alphabet, performed by 16 different subjects wearing coloured gloves. We collected a total of 11040 images. Each subject performed the 23 letters an average of 3 times. During the performance of each letter, a video sequence of approximately 10 image frames was recorded and labeled in order to test the performance of our system when classifying hand images from continuous video shots. When performing a particular sign, subjects followed visual instructions from official Irish Deaf Society materials with the aim of ensuring natural performance of postures. A random selection of the images were validated by a certified Irish sign language teacher to ensure subjects had performed signs correctly.

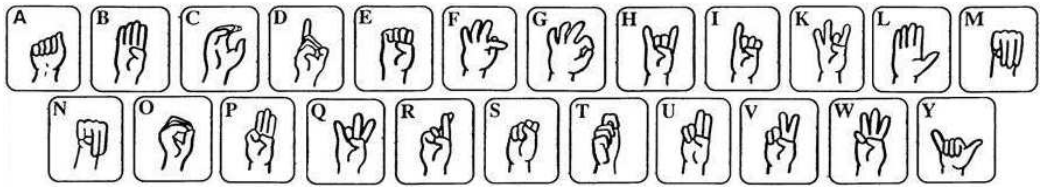


Figure 3.8: 23 Static Letters of the ISL Alphabet (The signs for "j", "x" and "z" cannot be performed statically and were not further considered)

All hand posture images were recorded with subjects asked to perform

the postures naturally. Due to the natural performance of the hand postures, there was a large variance in the type of hand postures performed for each sign. Variations in performance were only limited by that of sign language limitations (i.e. a large variation in orientation may give a posture a different meaning and thus was not allowed, as instructed by a certified Irish sign language translator). Figure 3.9 shows a visual example of a number of different ways the 'D' sign was performed by different subjects. It can be seen that the even though all hand postures in Figure 3.9 have the same sign classification, there is a clear difference in sign performance due to interpersonal differences. This illustrates the difficultly in the problem of user independent hand posture recognition.

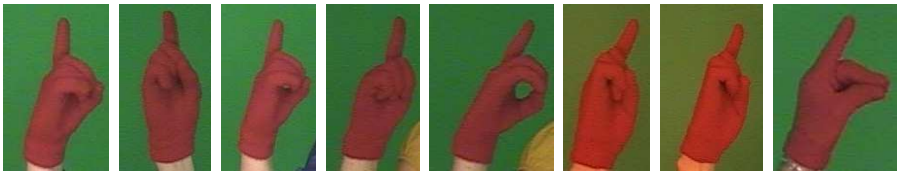


Figure 3.9: Example of variation in performance of the 'D' sign

In this data set, tracking of the hands is performed by tracking coloured gloves (see Figure 3.10(a)) using the Mean Shift algorithm [CRM00]. The key feature used for the analysis of the hand shape is the external contour made by the hand (see Figure 3.10(c)). To extract the external contour of the hand we segment the glove region in the image (see Figure 3.10(c)). This segmentation is carried out using a back projection image computed during the Mean Shift algorithm. Each pixel in the back projection image

represents the color similarity between the pixel and the predefined glove color (see Figure 3.10(b)). After segmenting the hand region, the external contour of the hand blob is extracted using a border following algorithm [Sb85].

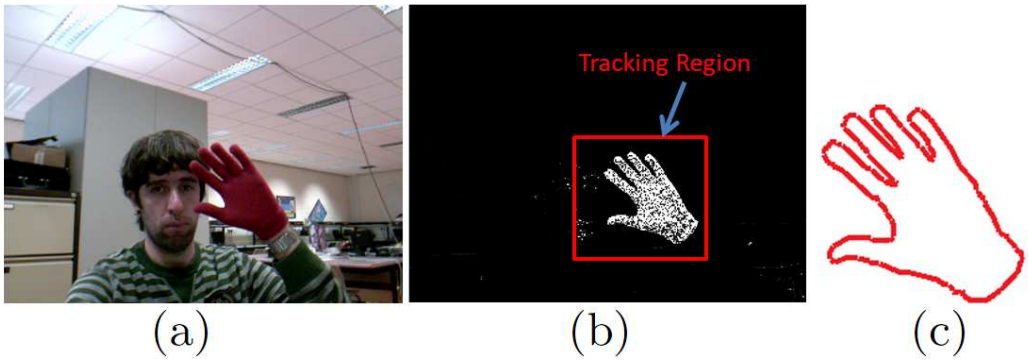


Figure 3.10: Feature Extraction from Image (a)Original Image (b)Back Projection Image (c)Extracted Contour

The back projection image, which is used to find the hand contour in an image, can typically contain varying levels of noise. The noise in a back projection image refers to segmentation noise where white pixels are not part of the hand region, or where black pixels are part of the hand region. Noise in a back projection image can produce hand contours which hold noise. Figure 3.11 shows an example of some noisy back-projection images, and the corresponding contours extracted from the images, which were used during experiments discussed in later Sections.

In our experiments, the system's ability to deal with noise is tested due to the presence of typical segmentation noise in the back projection images.

Figure 3.11: Example of Noisy Back Projection Images and Corresponding Noisy Contours

3.5 Evaluation of Discriminatory Properties

In this section we discuss experiments conducted on different variations of Size Functions in order to find the features which best discriminate between hand postures. We also perform experiments to evaluate the discriminatory properties of combining Hu Moments with our Size Function features and examine whether or not these features offer complementary information about hand posture patterns.

3.5.1 Size Functions and PCA performance

To examine how the eigenspace representation of a Size Function performs at discriminating between correct and incorrect signs performed by different people, an experiment is carried out to compare the eigenspace Size Function and the unmodified Size Function representations.

We evaluate our proposed features on the Jochen Triesch Static Hand Posture data set and on the ISL data set. For each hand sign in the data set we store a single hand image as a control image. The remaining set of hand images are stored as test images. We evaluate our proposed features by computing the distance between each control Size Function representation

and all test contour Size Function representations.

We generate Receiver Operating Characteristics (ROC) graphs for each of the Size Function representations by calculating a confusion matrix from the control contour and test shape distance comparisons. See Appendix B.1 for a detailed description of ROC analysis and Area Under the Curve (AUC) performance measures.

We define the function $D(C_k,C_l)$ as the distance measure computed between the control contour C_k and the test contour C_l. This procedure is carried out for both the eigenspace Size Function and the unmodified Size Function representations. To generate multiple points on the ROC graph, a confusion matrix is calculated from different threshold values T $(0 \leq T < +\infty)$.

Firstly, we compute a ROC graph for the unmodified Size Function representation. For each contour C, we compute the set of Size Functions $\Gamma_\varphi(C)$. The distance between two unmodified Size Functions is then calculated by a Euclidean distance measure according to Equation 3.5. Results of this test show an AUC measurement of 0.735 and 0.756 for the ISL data set and the Triesch data set respectively.

$$D^{'}(C_k, C_l) = \sqrt{\sum_{\theta=0}^{2\pi} \sum_{i=0}^{N} \sum_{j=0}^{N} (\Gamma_\theta(C_k)[i,j] - \Gamma_\theta(C_l)[i,j])^2} \tag{3.5}$$

A ROC graph is then computed for the eigenspace Size Function representation. For each contour C, we compute the set of Size Functions $\Gamma_\varphi(C)$ and then calculate the eigenvectors and eigenvalues for that set of Size Functions. For each θ, we choose only the first P eigenvectors, resulting in a set of eigenvectors $M_\theta(C)$, with dimensions $N \times P$, and a set of eigenvalues

$W_\theta(C)$ of dimension P. The eigenspace Size Function distance between two contours is then calculated by a Euclidean distance measure according to Equation 3.6. Results of this test produced an AUC measurement of 0.789 and 0.801 for the ISL data set and the Triesch data set respectively.

$$D^{\neg}(C_k, C_l) = \sqrt{\sum_{\theta=0}^{2\pi} \sum_{p=0}^{P} \sum_{i=0}^{N} (M_\theta(C_k)[p,i] - M_\theta(C_l)[p,i])^2} \tag{3.6}$$

A further modification to the eigenspace Size Function representation is proposed. We propose a scaling of the eigenvectors of each eigenspace Size Function based on a variance measure of their associated eigenvalues.

We define our weighted eigenspace Size Function $\zeta_\theta(C)$ in Equation 3.7.

$$\zeta_\theta(C)[p,i] = M_\theta(C)[p,i] \times \varpi_\theta(C,p) \times \varrho_\theta(C) \tag{3.7}$$

Where $\varpi_\theta(C,x)$ is a weighting factor for each eigenvector x associated with the Size Function indexed by θ according to Equation 3.8.

$$\varpi_\theta(C,x) = \frac{W_\theta(C)[x]}{\sum_{k=0}^{K} W_\theta(C)[k]} \tag{3.8}$$

The second weighting factor, $\varrho_\theta(C)$, is calculated for each set of eigenvectors associated with the Size Function indexed by θ, such that the Size Function with the greatest total variance is assigned a greater weighting according to Equation 3.9.

$$\varrho_\theta(C) = \frac{\sum_{p=0}^{P} W_\theta(C)[p]}{\sum_{\theta=0}^{N_\Theta} \sum_{p=0}^{P} W_\theta(C)[p]} \tag{3.9}$$

The distance, $d^{SF}(C_k,C_l)$, between the hand contours C_k and C_l is then defined as the Euclidian distance between the weighted eigenspace Size Functions:

$$d^{SF}(C_k, C_l) = \sqrt{\sum_{\theta=0}^{2\pi}\sum_{p=0}^{P}\sum_{i=0}^{N}(\zeta_\theta(C_k)[p,i] - \zeta_\theta(C_l)[p,i])^2} \tag{3.10}$$

A ROC analysis of the proposed weighted eigenspace Size Function shows an AUC measurement of 0.809 and 0.823 for the ISL data set and the Triesch data set respectively. The results of the experiment on the ISL data set show the weighted eigenspace Size Function has a total improvement of 7.4% when compared to the unmodified Size Function while results of the experiment on the Triesch data set show a total improvement of 6.7%. Figure 3.12 shows the ROC graphs associated with the AUC measurements reported above.

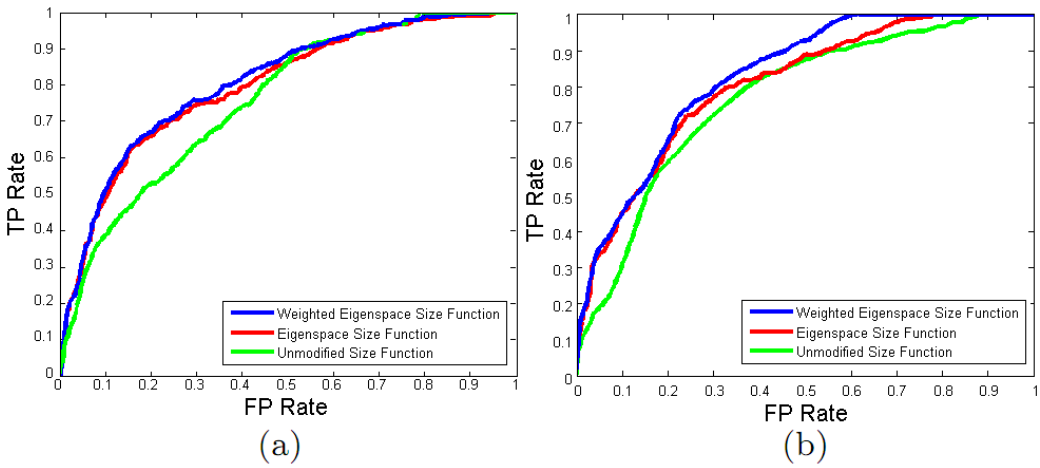


Figure 3.12: ROC graphs of weighted eigenspace Size Function, eigenspace Size Function and unmodified Size Function representations for (a) ISL data set and (b) Triesch data set

3.5.2 Hu Moments Performance

Along with the eigenspace Size Function representation of a hand, Hu moments of a segmented binary hand image are used as a feature to describe the posture of a hand. To test the suitability of Hu moments as a hand posture feature, a similar experiment to the one described in Section 3.5.1 is conducted. Hu moments are extracted from the control and test images, described in Section 3.5.1. The distance between each of the control Hu Moments and test Hu Moments is calculated from the total absolute difference between each augmented moment described in Equation 3.11. Where the augmented moment is a metric implemented in the OpenCV library [IC00] and is described in Equation 3.12.

$$d^{Hu}(C_k, C_l) = \sum_{i=1}^{7} \left|\Lambda\big(I\left(C_k\right)[i]\big) - \Lambda\big(I\left(C_l\right)[i]\big)\right| \tag{3.11}$$

$$\Lambda(hu) = \frac{1}{sign(hu) \times \log(hu)} \tag{3.12}$$

A ROC graph is then generated for the Hu moments using a method similar to the method described in Section 3.5.1. The AUC for the ROC graph is 0.796 and 0.852 for the ISL data set and the Triesch data set respectively.

3.5.3 Combining Size Function and Hu Moments

The ultimate goal the work discussed in this chapter is to find the best possible classification scheme for recognising hand postures. We have shown that both the eigenspace Size Functions and Hu moments features can sufficiently

discriminate between positive and negative examples with an AUC of 0.809 and 0.796 respectively for the ISL data set and an AUC of 0.823 and 0.852 respectively for the Triesch data set.

It is suggested that the combination of different classifier designs can potentially offer complementary information about the patterns to be classified. This complimentary information could then be harnessed to improve the performance of the selected classifier [KHDM98]. The classification system designed in this work uses a combination of different features, therefore a measure of the performance of the combination of eigenspace Size Functions and Hu moments is carried out.

The boolean expression defined in Equation 3.13 is used to determine the combined classifier's output. A true result corresponds to the classifier predicting that hand contours C_k and C_l are of the same hand posture category.

$$\psi(C_k, C_l) = d^{Hu}(C_k, C_l) \leq T_{hu} \cap d^{SF}(C_k, C_l) \leq T_{sf} \tag{3.13}$$

To examine the combined performance of the two measurements, an exhaustive grid search on the threshold values T_{hu} and T_{sf}, the threshold for the Hu moments measurement and the threshold for the Size Function measurements respectively, is carried out. For each $\{T_{hu}$,$T_{sf}\}$ a confusion matrix was computed on the output of Equation 3.13 when applied to the set of hand images used in Sections 3.5.1 and 3.5.2. We then choose the confusion matrix with the best True:False ratio (TF_{Ratio}) described in Equations 3.14-3.16.

$$TP_{Rate} = \frac{TP}{TotalPositives} \tag{3.14}$$

$$FP_{Rate} = \frac{FP}{TotalNegatives} \tag{3.15}$$

$$TF_{Ratio} = \frac{TP_{Rate} + (1 - FP_{Rate})}{2} \tag{3.16}$$

For both data sets, results of the grid search show that the best True:False ratio computed was better than that of any points on the ROC graphs computed from the individual Size Function features and Hu moment features respectively. Figure 3.13 shows the ROC graphs from the experiment and Table 3.1 details the best True:False ratios for the different features. It can be concluded from the results of this experiment that, based on the data from both data sets, the combination of the eigenspace Size Function and Hu moment representations provide complementary information about the shape of a hand.

Table 3.1: Best True:False Ratios

	ISL Data Set	**Triesch Data Set**
Best Combined TF_{Ratio}	0.797	0.794
Best Size Function TF_{Ratio}	0.733	0.75
Best Hu Moment TF_{Ratio}	0.764	0.77

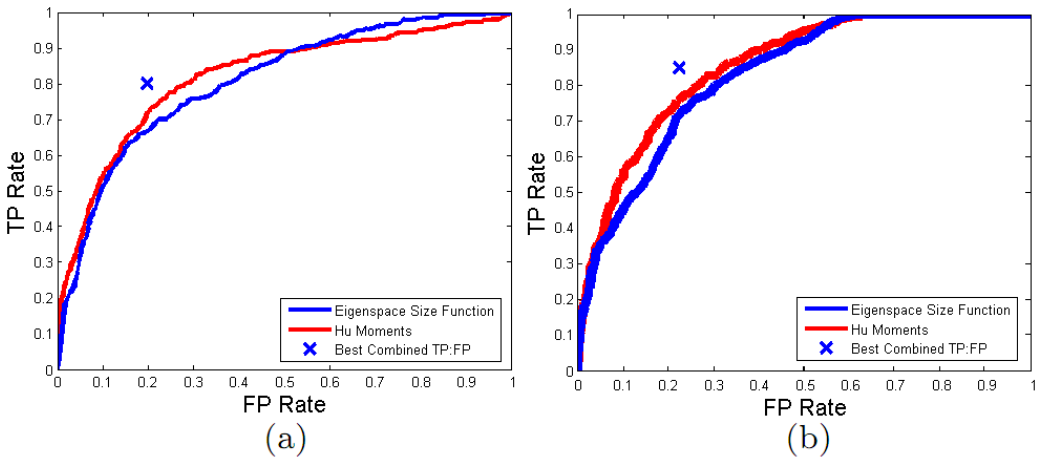


Figure 3.13: ROC graph of combined features for (a) ISL data set and (b) Triesch data set

3.5.4 Size Function Parameters

A ROC analysis is performed to select the optimal combination of parameters (N,N_Θ,P), the size of the Size Function, the number of graph rotations and the number of principal components respectively.

The same process described in Section 3.5.1 is carried out to evaluate the performance of the Size Function using different values of (N,N_Θ,P) where ($2 \leq N \leq 32$), ($2 \leq N_\Theta \leq 16$) and ($1 \leq P \leq N$).

It should be noted that as N increases, the margin between $G_{\varphi \leq y}$ and$G_{\varphi \leq x}$, the graphs used to calculate the values of the Size Function, decreases. As the margin decreases, smaller variations in the measuring function are identified as separate connected components. Therefore, as N increases, the Size Function becomes more sensitive to small changes in shape and noise. As N decreases, the Size Function become less sensitive to large shape variations

and therefore performs poorly at discriminating between signs.

Since the data collected in this work uses real sign data, typical segmentation noise can be present in the extracted contours. The existence of noise makes finding an optimal value of N an important goal as we must find an N which is not sensitive to noise but can discriminate between signs.

During the performance evaluation of the different parameters (N,N_Θ,P), we also found that adjusting P varied the system's sensitivity to noise. The more principal components used, the more sensitive the system becomes to noise. We conclude from this that the principal component holds the main information about the hand posture, while the lower components hold information about small variations in the contour shape.

Table 3.2: Parameter Combination AUC

(N, N_Θ, P)	ISL Data AUC	Triesch Data AUC	(N, N_Θ, P)	ISL Data AUC	Triesch Data AUC
(16,6,1)	**0.809**	**0.823**	(16,6,2)	0.794	0.811
(16,4,1)	0.791	0.822	(16,4,2)	0.783	0.804
(16,8,1)	0.799	0.814	(16,8,2)	0.786	0.792
(8,6,1)	0.782	0.801	(8,6,2)	0.779	0.787
(8,4,1)	0.780	0.796	(8,4,2)	0.769	0.783
(8,8,1)	0.791	0.807	(8,8,2)	0.773	0.792

Table 3.2 details different ROC AUCs for different parameter combinations computed from the ISL data set and the Triesch data set. Although experiments are carried out exhaustively on different parameter combinations, we present only the 12 best parameter combinations. Parameters within the bounds; ($2 \leq N \leq 32$), ($2 \leq N_\Theta \leq 16$) and ($1 \leq P \leq N$), do not have a significant effect on the AUC with the lowest AUCs being 0.681 and 0.693 for our

ISL data set and the Triesch data set respectively. The parameter combination which produced the best AUC was N= 16, N_Θ= 6 and P= 1. This was the parameter combination used in all tests described in Sections 3.5.1 and 3.5.3 above, and will be used for the posture recognition techniques which we will discuss in Section 3.6.

3.6 Recognition Framework

In Section 3.5 we have shown that our eigenspace Size Functions and Hu moments possess strong hand shape discriminatory properties. We now describe our user independent framework for recognising hand postures using these shape representations.

A set of SVMs [CL01] are trained on data, using the discussed shape representations, extracted from labeled images.

Given an unknown hand image, the relevant features are extracted and the SVMs use the data to estimate the most probable hand posture classification.

To classify an unknown hand posture C, it must be assigned to one of the Y possible posture classes $(\alpha_1, \alpha_2, ..., \alpha_Y)$. The proposed recognition framework uses two distinct measurement vectors to represent a hand posture. For each posture class, α_y, a set of two SVMs, $\{SVM_y^{sf}, SVM_y^{hu}\}$, are used to calculate $P(\alpha_y|I(C),\zeta(C))$, the probability that posture C belongs to class α_y given measurement vectors $I(C)$ and $\zeta(C)$. Where $I(C)$ is the set of Hu Moments and $\zeta(C)$ is the weighted eigenspace Size Function calculated from hand contour C.

3.6.1 Support Vector Machines

Support Vector Machines are a set of supervised learning methods used in classification and regression. A one against all Support Vector Machine (SVM) model is used in this work, and training of the SVM consists of providing the SVM with data for two classes (see Appendix A.2 for detailed discussion on implementation of SVM training and classification). Data for each class consists of a set of n dimensional vectors. An Radial Basis Function (RBF) kernel is applied to the data and the SVM then attempts to construct a hyper plane in the n-dimensional space, attempting to maximise the margin between the two input classes.

The SVM type used in this work is C-SVM using a non linear classifier by means of the kernel trick as proposed by Aizerman et al. [ABR64]. The kernel used is a RBF kernel as defined by $K(x, x') = exp(-\gamma\|x - x'\|^2)$ (see Appendix A.2.1).

We utilise an extension to SVMs in order to obtain probability estimates for each class. This is carried out by using a technique proposed by Platt [PP99] (see Appendix A.2.2).

3.6.2 Training

Given a training set of hand images consisting of multiple labeled images of each hand posture class we train a set of SVM classifiers as follows:

Weighted eigenspace Size Function data and Hu moment data are extracted from the training set images to create the matrices $H_y = (I_{y1}, I_{y2}, ..., I_{yL})$ and $\Psi_y = (\zeta_{y1}, \zeta_{y2}, ..., \zeta_{yL})$ where L is the total number of training images

recorded for each posture class α_y.

To train each SVM_y^{sf}, the matrix Ψ_y is used as the positive labeled training data and $\overline{\Psi_y} := (\Psi_j)_{j \neq y \cap j \epsilon \{1..Y\}}$ is used as the negative labeled training data. Similarly, each SVM_y^{hu} is trained using H_y as the positive labeled data and $\overline{H_y} := (H_j)_{j \neq y \cap j \epsilon \{1..Y\}}$ as the negative labeled data. The support vector machines SVM_y^{sf} and SVM_y^{hu} are then trained to maximise the hyperplane margin between their respective classes $(\Psi_y, \overline{\Psi_y})$ and $(H_y, \overline{H_y})$.

There are two parameters while using RBF kernels: C and γ. V-fold cross validation was carried out to compute optimal values for C and γ.

3.6.3 Posture Classification

To classify an unknown posture C, each SVM_y^{sf} and SVM_y^{hu} will calculate $P(\alpha_y|\zeta(C))$ and $P(\alpha_y|I(C))$, the probability $\zeta(C)$ and $I(C)$ belong to class α_y respectively. Classifier weights, μ_y^{sf} and μ_y^{hu}, used to determine the overall probability, are calculated by Equations 3.17 and 3.18, where cv_y^{sf} and cv_y^{hu} are the cross validation accuracies achieved during the training of each SVM_y^{sf} and SVM_y^{hu} respectively.

$$\mu_y^{sf} = \frac{cv_y^{sf}}{cv_y^{sf} + cv_y^{hu}} \tag{3.17}$$

$$\mu_y^{hu} = \frac{cv_y^{hu}}{cv_y^{sf} + cv_y^{hu}} \tag{3.18}$$

A weighted combination of the probabilities is then calculated to generate the overall probability $P(\alpha_y|I(C),\zeta(C))$ according to Equation 3.19.

$$P(\alpha_y|I(C),\zeta(C)) = (P(\alpha_y|\zeta(C)) \times \mu_y^{sf}) + (P(\alpha_y|I(C)) \times \mu_y^{hu}) \quad (3.19)$$

3.6.4 Experiments

We evaluate our recognition system using both data sets discussed in Section 3.4.

For the ISL data set, we train the SVMs on 5520 hand posture images. The 5520 training images were comprised of data from 8 of the 16 subjects used. We then test our recognition framework on the remaining 5520 images.

For the Triesch data set we carry out two evaluation protocols (P1 and P2). We first perform an evaluation based on the same protocol as Triesh et al. [TvdM02]. We train the SVMs on each of the 10 hand signs using data extracted from 3 of the 24 signers. The system is then tested on all hand signs from the remaining 21 subjects. The second evaluation protocol we perform is based on the work of Just et al. [JRM06], where 8 signers are used for training a validation and the remaining 16 are used for testing.

We carry out the following tests for the ISL data set and both of the Triesch evaluation protocols: For each posture $C_i \epsilon \{1...L\}$, where L is the total number of images in the test set, the classification probabilities $\Delta_i := [P(\alpha_1|I(C_i),\zeta(C_i)), ..., P(\alpha_Y|I(C_i),\zeta(C_i))]$ is calculated. To test the performance of the system a ROC analysis is carried out on the classification of the test images. For each posture class $y \epsilon \{1...Y\}$ a confusion matrix is calculated. To generate multiple points on the ROC graph, a confusion matrix

is calculated from different threshold values T ($0 \leq T < 1$). Table 3.3 details the AUC of the ROC graph generated from the classification of both the training data and the test data of the ISL data set. It can be seen from the AUC measures, generated from both the training data and test data, that each classifier performs robustly at classifying each hand shape.

The results also show that the overall recognition rate of the test set is within 1.6% of the overall classification of the training set. Since the subjects used to record the test data were different to the subjects used to record the training data, it can be concluded from these results that the proposed hand posture recognition framework performs well at recognising hand postures independent of the subjects performing the postures. Table 3.3 also shows the classifier weights, μ_y^{sf} and μ_y^{hu}, calculated during the cross validation stage of the training and used in the recognition experiments. These weights give an indication of which classifier performs best at classifying the different hand shapes. As can be seen from Figure 3.8, the hand shape for "A" and "E" are quite similar, as are the hand shapes for "F" and "G". From the classifier weights, we see that the Hu moments have a greater discrimination when classifying shapes with little contour variation, such as the signs for "A" and "E", whereas the eigenspace Size Function has a greater discrimination when classifying signs with a larger variation in the contour shape, such as the signs for "F" and "G".

Table 3.3: Classification AUC Performance

Letter	**Training Set Recognition**	**Test Set Recognition**	**Hu Moment Weighting μ_c^{hu}**	**Size Function Weighting μ_c^{sf}**
A	0.996	0.995	0.57	0.43
B	0.997	0.989	0.58	0.42
C	0.930	0.907	0.49	0.51
D	0.993	0.971	0.54	0.46
E	0.991	0.996	0.57	0.43
F	0.976	0.991	0.46	0.54
G	0.993	0.974	0.48	0.52
H	0.995	0.967	0.48	0.52
I	0.980	0.971	0.50	0.50
K	0.991	0.934	0.47	0.53
L	0.984	0.932	0.48	0.52
M	0.985	0.935	0.55	0.45
N	0.999	0.954	0.52	0.48
O	0.999	0.930	0.54	0.46
P	0.999	1.000	0.58	0.42
Q	0.962	0.989	0.45	0.55
R	0.992	0.960	0.51	0.49
S	1.000	1.000	0.56	0.44
T	1.000	1.000	0.56	0.44
U	0.976	0.993	0.52	0.48
V	1.000	1.000	0.49	0.51
W	1.000	1.000	0.45	0.55
Y	1.000	1.000	0.50	0.50
Mean	**0.989**	**0.973**	**0.52**	**0.48**

Results of the evaluation on the Triesch data set, detailed in Table 3.4, show that the Elastic Graph matching algorithm of Triesch et al. [TvdM02] achieves a higher recognition rate than our method. The small amount of training data would seem to contribute to the lower recognition rate of our method. Results show that as the training set grows, from 3 subjects to 8 subjects, the recognition rate increases by 6.4%. Our method also shows a

better recognition rate than the work of Just et al. [JRM06], when trained on data from 8 subjects. Flasinski et al. [FM10] conduct experiments on a subset of the Triesch Data set and report a recognition rate of 85.4% when recognising 144 hand postures images. Flasinski et al. also reported that the average computation time for classifying a single image was 0.13 seconds on a computer with a 1.83Ghz Intel Core 2 CPU.

Table 3.4: Recognition Performance

	#Training	#Test	Number	Correct	Percentage	AUC
Our Method P1	3	21	418	356	85.1	0.827
Triesch et al.	3	21	418	392	95.2	-
Our Method P2	8	16	320	294	91.8	0.935
Just et al.	8	16	-	-	89.9	-
Flasinski et al.	-	-	144	123	85.4	-

3.6.5 Continuous Recognition

To show an example of the functionalality of our system when classifying signs from a video stream, we show results of a video based recognition experiment. It should be noted that the purpose of the experiment discussed in this Section is to show a qualitative example of posture classification and does not serve as a full system evaluation. We utilise a live gesture feedback application, which we will discuss in Chapter 5, along with the SVMs, trained on the Triesch data set using protocol 2, to classify signs from a continuous video sequence. In the video sequence, an unseen user performs each of the 10 signs one after the other. In the first frame of the video, the hand position is manually selected and a skin colour histogram of the hand is recorded. For each successive frame, the mean shift algorithm is used to locate the

hand region [CRM00]. We then perform Canny edge detection on the hand region, followed by a dilation operation, and the contour is then extracted from the edge detected image using a border following algorithm [Sb85](see Figure 3.14). Figure 3.15 illustrates the results of the video based posture classification, where the graph depicts the probability for each hand posture class for each image frame. It can be seen from the ground truth labels on the graph that our system performs well at classifying postures in each image frame.

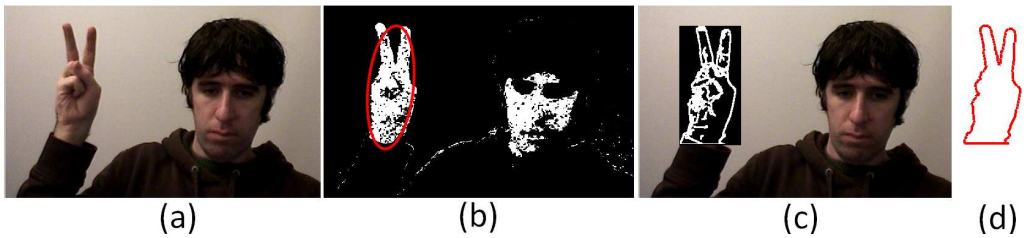


Figure 3.14: Feature Extraction for Continuous Recognition Experiment (a) Original Image (b) Skin Colour Segmentation using Mean Shift Algorithm (c) Edge Detected Hand Region (d) Extracted Contour

3.7 Conclusion

This main contribution of the work detailed in this chapter is a user independent hand shape feature, a weighted eigenspace Size Function, which is a strong improvement over the original Size Function feature. We implement a user independent hand posture recognition framework using our weighted eigenspace Size Function and a set of Hu moments, which we show to complement our proposed feature.

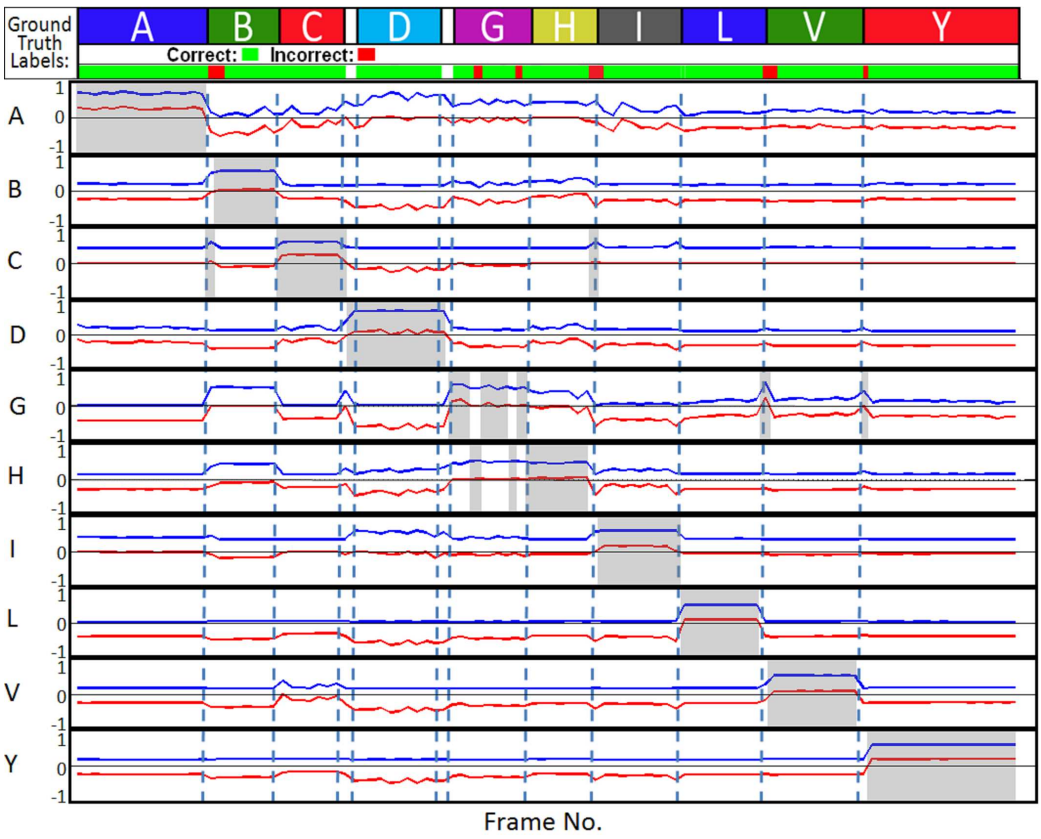


Figure 3.15: Recognition probabilities for Continuous Video Stream. For each frame, the classifier which outputs the maximum likelihood is denoted as the grey area. For each classifier output we denote the blue plot as the likelihood output of that classifier, while the red plot denotes the difference between the classifier output and the likelihood of the classifier with the second highest likelihood (Thus, for the red plot, values above 0 denotes the maximum likelihood, values equal to 0 denotes the second highest likelihood and values below 0 denotes all other likelihoods).

Our eigenspace Size Function performs significantly better at discriminating between different hand postures than the unmodified Size Function when tested on two different user independent hand posture data sets. An increase in performance of 7.4% and 6.7% is shown for our weighted eigenspace Size Function when compared to the unmodified Size Function using a simple Euclidian distance classifier. We propose a user independent, SVM based, recognition framework using a combination of our weighted eigenspace Size Function and Hu Moments. Results of a user independent evaluation of the recognition framework show our framework has a ROC AUC of 0.973 and 0.935 when tested on the ISL data set and the Treisch data set respectively. These improvement suggests that the principal component of the Size Function holds the main information about the hand posture, while the lower components hold information about small variations in the contour shape that are specific to a particular person.

The advantages of our system, in contrast to other hand posture recognition systems described in Section 2.2.3, is that our system accurately recognises hand postures independent of the person performing them and independent of the style of the posture being performed. Our system also performs this person independent recognition in real time from low resolution images of the hand taken from images where the FOV includes the full upper body. These are a significant set of advantages as the combination of these advantages makes our hand posture framework an ideal system to be used in full sign language sentence level recognition systems.

Chapter 4

Spatiotemporal Gesture Recognition [1]

4.1 Introduction

Recognising gestures which appear in sign language is a challenging problem. An accurate recognition system must be able to distinguish between meaningful sign segments and inter sign transitions. Each meaningful sign lacks a clear categorical structure and signs which have the same meaning can occur at different timescales. Inter-gesture transitions also occur between these meaningful signs where, for example, when performing hand gestures the hands must move from the end point of the previous gesture to the start

[1] Various aspects of the work described in this chapter have been published in a number of conference proceedings. The completed work has been accepted for a book chapter publication: **D. Kelly**, J. Mc Donald and C. Markham,"Recognition of Spatiotemporal Gestures in Sign Language using Gesture Threshold HMMs", In Machine Learning for Vision Based Motion Analysis, Springer LNCS, In Submission

point of the next gesture. These inter-gesture transition periods are called movement epenthesis [SR89] and are not part of either of the signs. The difficulty in developing gesture recognition techniques is that the recognition framework must be able to classify meaningful signs segments while also identifying movement epenthesis.

As discussed in Section 2.2.2, the limitation of current methods of continuous gesture recognition is that explicit training of movement epenthesis models are required, explicit rules for gesture segmentation must be specified or unnatural constraints are put on the signer, such as unnatural pauses between words. The main contribution of the work detailed in this chapter is that we propose a Gesture Threshold Hidden Markov Model (GT-HMM), which is a spatiotemporal gesture recognition framework which does not require explicit epenthesis training or specific rules to determine gesture boundaries.

4.1.1 Chapter Outline

Before discussing our GT-HMM framework, we give an overview of HMMs and Threshold HMMs in Sections 4.2 and 4.3 respectively. In Section 4.4, we then detail the implementation of our proposed GT-HMM framework which is specifically designed to classify manual and non-manual signals and to identify movement epenthesis. In Section 4.5, we carry out a comprehensive evaluation of the GT-HMM framework and compare with current state of the art temporal event modeling frameworks such as CRFs, HCRFs, LDCRFs and standard HMM systems. Since sign language involves not only hand

gestures but also non-manual signals, gesture recognition evaluations will be carried out on hand gestures and non-manual signals conveyed through head and eye brow movements.

4.2 Hidden Markov Models

HMMs are a type of statistical model that can model spatiotemporal information in a natural way. HMMs have efficient algorithms for learning and recognition, such as the Baum-Welch algorithm and Viterbi search algorithm [Rab89]. They have been utilised for the task of gesture recognition in a large number of works in the literature. HMMs where first used for the task of gesture recognition by Yamato et al. [YOI92] and for the task of sign language recognition by Starner et al. [SPW98]. In these seminal works, the authors state that the key characteristics of HMMs, which make them suitable for gesture recognition, is their learning ability and time-scale invariability.

A HMM is a collection of states connected by transitions. Each transition (or time step) has a pair of probabilities: a transition probability (the probability of taking a particular transition to a particular state) and an output probability (the probability of emitting a particular output symbol from a given state).

A gesture sequence is represented as a set of observations. An observation $\mathbf{f}_t$, is defined as an observation vector made at time t, where $\mathbf{f}_t$= $\{o_1, o_2, ..., o_M\}$ and M is the dimension of the observation vector. A particular gesture sequence is then defined as G= $\{\mathbf{f}_1,\mathbf{f}_2, ...,\mathbf{f}_T\}$. We will discuss the features used to represent different gesture types in Section 4.5.

A HMM is characterised by the following:

1. N, the number of states in the model. We denote the individual states as S= $\{s_1$,s_2, ...,$s_N\}$, and the state at time t as q_t.

2. M, the dimension of the observation vector.

3. $A = \{a_{ij}\}$, the state transition probability distribution. Where A is an $N \times N$ matrix and a_{ij} is the probability of making a transition from state s_i to s_j.

4. $B = \{b_j(\mathbf{f})\}$, the observation symbol probability distribution. Where b_j is the probability distribution in state j and $1 \leq j \leq N$.

5. $\pi = \{\pi_i\}$, the initial state distribution.

The compact notation λ= $\{A, B, \pi\}$ is used to indicate the complete parameter set of the model where A is a matrix storing transition probabilities a_{ij} between states s_i and s_j, B is a matrix storing output probabilities for each state and π is a vector storing initial state probabilities.

HMMs can use either a set of discrete observation symbols or they can be extended for continuous observations signals. In this work we use continuous multidimensional observation probabilities calculated from a multivariate probability density function.

The observation probability is expressed in the form shown in Equations 4.1 and 4.2:

$$b_j(\mathbf{f}) = \aleph(\mathbf{f}; \mu_j, \Sigma_j) \quad (4.1)$$

$$= (2\pi)^{-\frac{N}{2}} |\Sigma|^{-\frac{1}{2}} exp\left(-\frac{1}{2}(\mathbf{f}-\mu_j)^T \Sigma_j^{-1}(\mathbf{f}-\mu_j)\right) \quad (4.2)$$

Where $\mathbf{f}$ is the M dimensional vector being modeled, b_j is the vector probability in state j and $\aleph$ is a Probability Density Function (PDF) of an M-dimensional multivariate gaussian distribution, with mean vector μ_j and covariance Σ_j.

4.2.1 HMM Algorithms

Given the form of a HMM, there are three algorithms that can be performed on the HMM that make HMMs useful in real-world applications:

- The forward backward algorithm [Rab89] is used to calculate $P(G|\lambda)$, the probability of the observation sequence G= $\{\mathbf{f}_1,\mathbf{f}_2, ...,\mathbf{f}_T\}$ given the model λ= $\{A, B, \pi\}$.

- The Viterbi Algorithm [Rab89] is used to find the single best state sequence $Q = \{q_1, q_2, ..., q_T\}$, for the given observation sequence G= $\{\mathbf{f}_1,\mathbf{f}_2, ...,\mathbf{f}_T\}$.

- The Baum-Welch algorithm [Rab89] is used to determine the model parameters (A, B, π) to maximise the probability of the observation sequence given the model.

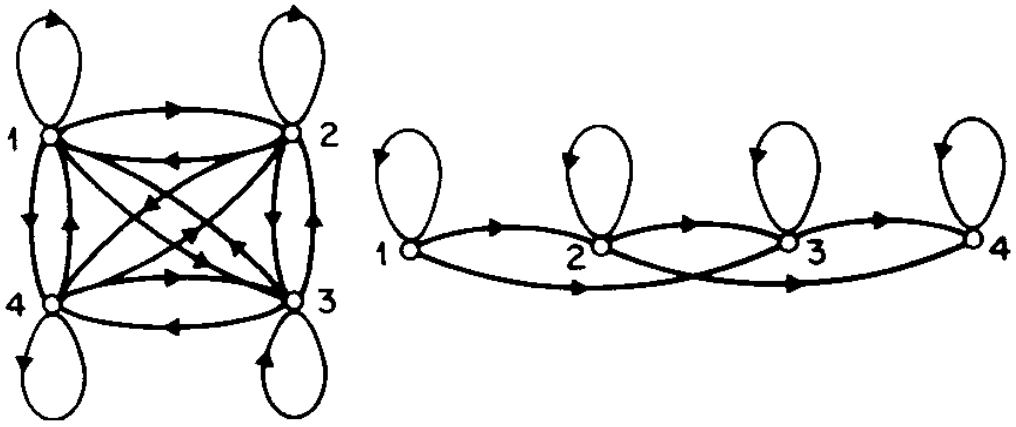
Appendix A.1 gives a detailed account of each of these algorithms.

4.2.2 Types of HMMs

There are two main types of HMMs:

1. Ergodic Model: A HMM in which every state of the model could be reached from every other state of the model (See Figure 4.1(a)).

2. Left-Right Model (Bakis Model): A HMM in which the state sequence associated with the model has the property that as time increases the states index increases or stays the same (i.e. the states progress from left to right), (See Figure 4.1(b)).



(a) Ergodic (b) Left-Right

Figure 4.1: HMM Model Types

The left-right model has the desirable property that it can readily model signals whose properties change over time. As a result, left-right models have been successfully applied to speech recognition tasks [Rab89] and more recently, to gesture recognition and sign recognition tasks [OR05].

4.3 Threshold HMM Model

For correct gesture spotting, the likelihood of a gesture model for a given pattern should be distinct enough. Unfortunately, although the HMM chooses a model with the best likelihood, we cannot guarantee that the pattern is really similar to the reference gesture unless the likelihood value is high enough. A simple thresholding for the likelihood often does not work. Therefore, Lee and Kim [LK99] proposed a Threshold HMM that yields the likelihood value to be used as a threshold. The threshold model was implemented to calculate the likelihood threshold of an input pattern and to provide a confirmation mechanism for provisionally matched gesture patterns. We build on the work carried out by Lee and Kim to create a framework for calculating a probability distribution of a two-handed input sign using continuous multidimensional observations. The computed probability distribution will include probability estimates for each pre-trained sign as well as a probability estimate that the input sign is a movement epenthesis.

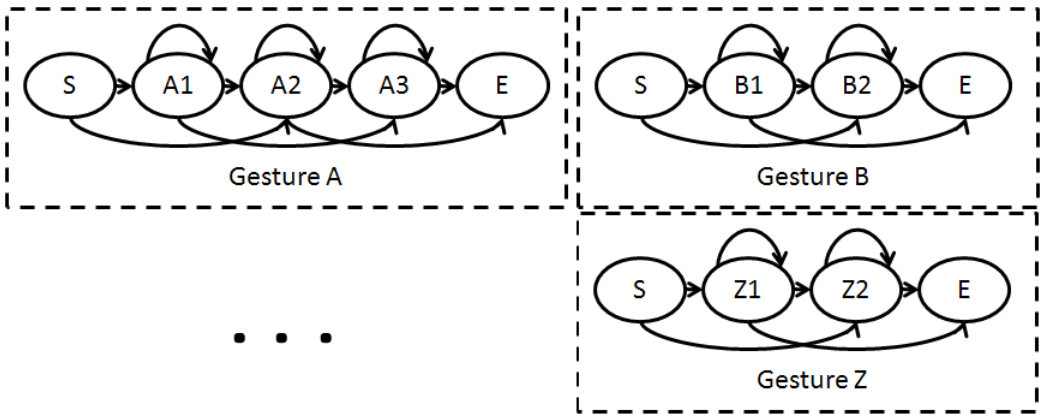
If a set of left-right HMMs can be trained such that each state represents a particular gesture segment, then a self transition of a state represents a particular segment of a target gesture and the outgoing state transition represents a sequential progression of the segments within a gesture sequence. With this in mind, an ergodic model, with the states copied from all gesture models in the system, can be constructed as shown in Figures 4.2(a) and 4.2(b). Figure 4.2(b) shows the threshold model as a simplified version of the ergodic model where dotted lines denote null transitions. We illustrate the threshold model in a simplified manor where the null transitions repre-

sent transitions where no actual observations occur. This means that, for example, a transition can occur between state $A1$ and $Z2$ in a single observation by making a transition from $A1$ to E, E to S and then S to $Z2$. In reality this transition would occur in a single step (i.e. $A1$ to $Z2$), but for illustration purposes we use the null transitions so as to not over complicate the diagram with a large number of transition labels.

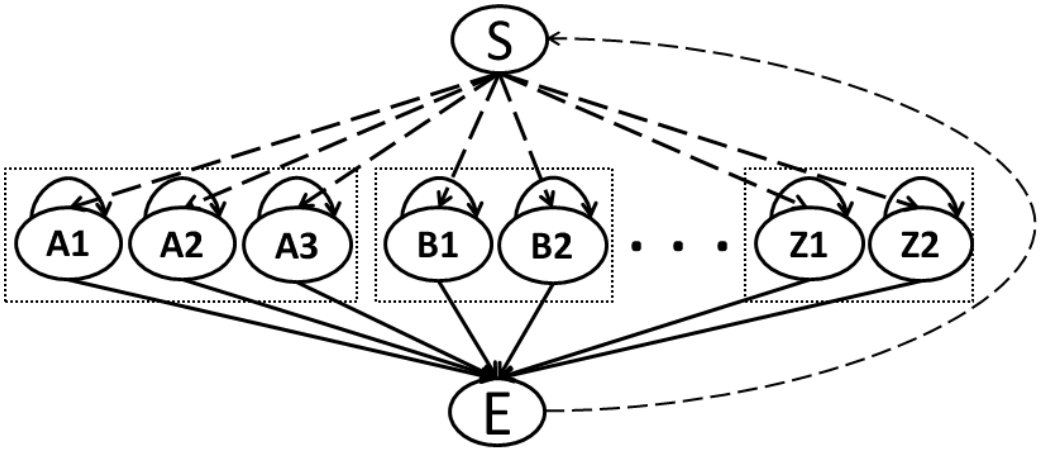
The threshold model states are created by copying all states from the left-right HMMs such that output observation probabilities and self transition probabilities are kept the same, but all outgoing transition probabilities are equally assigned as:

$$a_{ij} = \frac{1 - a_{ii}}{N - 1} \; \forall j, i \neq j \tag{4.3}$$

Where a_{ij} is the transition probability from state s_i to s_j and N is the number of states excluding the start and end states (The start and end states produce no observations). If each left-right HMM can be trained such that each state represents a gesture sub-pattern, then, by maintaining the self transition and output probabilities in the threshold states, a threshold model represents the set of all gesture sub-patterns. Constructing the threshold model as an ergodic structure thus makes it match well with all patterns generated by combining any of the gesture sub-patterns in any order. The likelihood of the threshold model, given a valid gesture pattern, would be smaller than that of the dedicated gesture model because of the reduced outgoing transition probabilities. However, the likelihood of the threshold model, given an arbitrary combination of gesture sub-patterns, would be



(a)



(b)

Figure 4.2: (a) Dedicated Gesture Models (b) Threshold Model (S- Start State, E- End State)

higher than that of any of the gesture models. Thus the threshold model, denoted as $\overline{\lambda}$, can be used as a movement epenthesis likelihood measure.

4.4 GT-HMM Framework

In this work, we propose a Gesture Threshold Hidden Markov Model (GT-HMM) as an improvement to the standard HMM and Threshold HMM, to automatically train and model natural gestures and movement epenthesis. The GT-HMM comprises of two main improvements to the standard HMM framework. Firstly, we develop a gesture subunit initialisation technique to create HMM states which model particular gesture sub-patterns. Secondly, we implement a threshold HMM, which utilises states which were initialised using the gesture subunits, to compute a dynamic epenthesis likelihood of input gestures. In this section we will detail the implementation of the GT-HMM.

4.4.1 GT-HMM Training

In order to create a robust threshold HMM which models movement epenthesis, dedicated HMM models must be trained such that each state represents a particular gesture segment. We develop a technique to do this by expanding on the Threshold HMM framework to develop a GT-HMM framework which utilises our proposed gesture subunit initialisation and training techniques. Our GT-HMM framework models continuous multidimensional gesture observations within a HMM network to recognise motion based gestures and identify movement epenthesis.

Each dedicated gesture model is trained on isolated gestures performed by a fluent signer. Before training a HMM using the Baum-Welch algorithm, the model must first be initialised. Initialisation includes the computation of an initial state transition matrix and calculation of each state's emission variables μ_j and Σ_j. In order to initialise these components of the HMM, an understanding of the gesture segmentation, or state transitions, must be developed. One approach to achieving this would be to explicitly hand label different subunits or gesture phonemes (signemes) [WSG02]. One of the overall goals of the work detailed in this thesis is to create a weakly supervised training and recognition system. Manual or automatic labeling is an integral step in creating valid sign data. Thus we envisage that all data will be labeled, or weakly supervised, by fluent signers. Since movement and position of the hands are two of the four building blocks of sign language which Stokoe [Sto05] identified, manually breaking these building blocks into smaller subunits would be an un-intuitive and time consuming step for fluent signers to label in a consistent manner. With this in mind, a training system was developed to initialise and train data with minimum human intervention where signs are labeled at a sign level and not at a signeme level.

Kim et al. [KC02] implemented an iterative HMM training procedure in order to estimate more accurate initial HMM parameters for automatic speech segmentation. In their method, hand labeled phoneme labels were used to initialise the HMMs. Following this, the Baum-Welch algorithm was run on the HMMs to tune the HMM parameters. The Viterbi algorithm was then run and the initialisation data and phoneme labels were realigned to

correspond with the Viterbi best paths. The HMM was then re-initialised using the realigned phoneme labels and this iterative procedure was repeated until no improvement was observed.

We extend this iterative HMM training model proposed by Kim et al. [KC02] to develop an iterative gesture subunit initialisation and training model. Our training model also includes an extra parameter selection layer which finds the best combination of (S,R), where S is the total number of states in the HMM and R is the reach of a state. In a left-right model, the reach is the number of states that it is possible to transition to from the current state. For a target sign, we extract data from a number of manually labeled isolated video sequences of a fluent signer performing that sign. Figure 4.3 shows a visualisation of isolated examples of the "Alot" sign extracted from video sequences.[2]

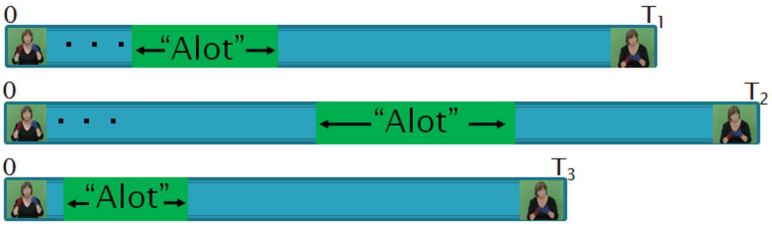


Figure 4.3: Visualisation of Isolated Examples of "Alot" sign Extracted from Video Sequences

The problem of automatically extracting isolated examples of a target sign using weakly supervised data will be discussed in Chapter 5.

[2] "Alot" is the Sign Language Gloss for the English language term "A large number of"

Gesture Subunit Initialisation

Extracting isolated examples of a sign produces a set of observation sequences $\Delta_y = \{G_y^1, G_y^2, ..., G_y^K\}$ where K is the total number of isolated examples. In HMM training, more accurate initial estimates of the HMM parameters produce more accurate classification results, as shown for example in [KC02]. The goal of our proposed initialisation technique is to automatically find gesture subunits and the most accurate initial emission variables which describe the gesture subunits. Our technique determines a labeling of the state that each observation vector most probably matches. These state labels are then used to determine the subset of observation vectors which are associated with a specific HMM state. The observation vectors within the subset can then be used to improve the estimate of the mean and covariance parameters associated with a state.

To initialise λ_y, the HMM which will model the sign indexed by y, we first calculate $S-1$ indices of each G_y^i which best segment the gesture into S sub-gestures. We propose an iterative temporal clustering algorithm to calculate the S sub-gestures. Figure 4.4 shows a visualisation of isolated examples of the "Alot" sign being segmented into $S = 3$ sub-gestures.

The key motivation for the development of our gesture subunit initialization technique was to find a way to best describe a gesture class in S segments, where each segment will be used to initialize each state in the HMM.

Time series segmentation aims to organize time series data into few intervals having uniform characteristics. Several high level representations of

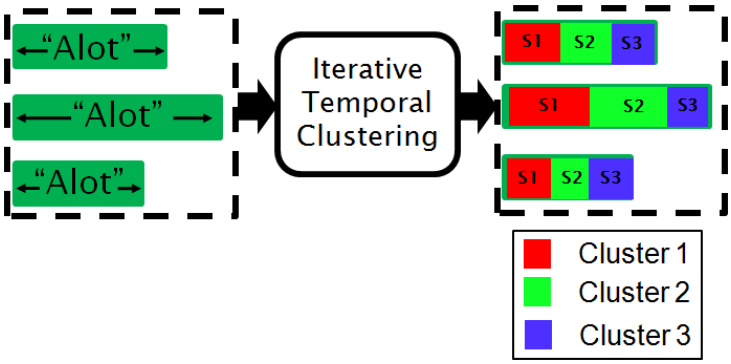


Figure 4.4: Initial Segmentation: Visualisation of Isolated Examples of "Alot" being Segmented into $S = 3$ sub-gestures

time series have been proposed, including Fourier Transforms [KCPM00], Wavelets [84799], Symbolic Mappings [84700] and Piecewise Linear Representation (PLR) [GS01]. In general, given a single time series T, these time series algorithms aim to produce the best representation of T using only K segments. In order to carry out time series segmentation for the purpose of HMM initialization, we must develop an algorithm which, given a set of time series $\mathbf{T} = \{T_1, ..., T_K\}$ with different lengths, can produce a single representation of the complete set of time series using K segments.

The objective of our temporal clustering algorithm is to cluster observations in a temporal structure such that each set of cluster indices, associated with each observation sequence, has a left to right clustering. As time increases, each cluster index should increase or stay the same. As an example, for S= 3, a cluster index sequence of '00110122' is considered an invalid temporal clustering. This index sequence is not a valid temporal clustering because the 5^{th} element in the sequence is less than the 4^{th} element and

thus does not have a left to right clustering. A cluster index sequence of '00011122' is considered a valid temporal clustering as each index is either greater than or equal to the index which occurred previously. Our iterative clustering algorithm attempts to cluster all observations, in the set of observation sequences Δ_y, such that all cluster index sequences have a valid temporal clustering.

Our temporal clustering algorithm is based on an iterative time scaling procedure which incorporates a time variable into the observation vectors in order to temporally segment each observation sequence. We define a time augmented observation vector $\mathbf{f}_t(\eta) = \{o_1, o_2, ..., o_M, \eta\}$ and a time augmented observation sequence $G^{\neg} = \{\mathbf{f}_1(\frac{1}{T}\Gamma^T), \mathbf{f}_2(\frac{2}{T}\Gamma^T), ..., \mathbf{f}_T(\Gamma^T)\}$, where Γ^T is a time scaling factor. Each iteration of the algorithm increases the time scaling factor and calculates cluster indices for each observation vector, using k-means clustering, until all time augmented observation sequences have a corresponding valid temporal clustering.

We use a sample set of nine sequences of the "Alot" sign in order to illustrate how our temporal clustering algorithm is used to initialise the HMMs.

As will be discussed in Section 4.5.3, the observation vectors used to describe a sign in this work are 5-dimensional vectors. In order to illustrate the temporal clustering of these observation vectors, we perform PCA on the data in order to reduce the dimensionality of the data to 1-dimension (see Figure 4.5). Although we illustrate the clustering results using the principal component, it is important to note that all clustering, including the results shown in this section, were calculated using the 5 dimensional observation

vector.

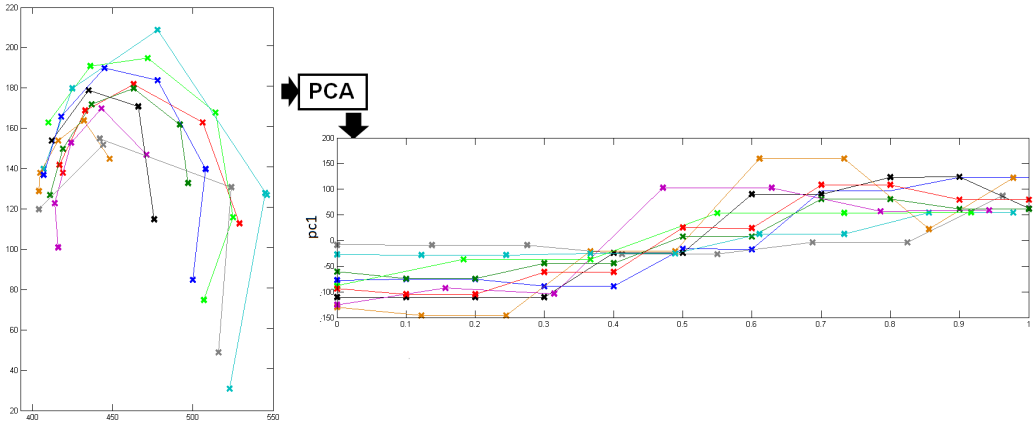


Figure 4.5: Visualisation of Principal Components of Sign Data for Left Hand (Left) 2 Dimensions, of the 5 dimensional feature vector, representing the x and y trajectories of the hand (Right) Principal component computed from the 5 dimensional feature vectors

We now illustrate an example of our temporal clustering algorithm applied to the left hand observation sequences for the sign "Alot" when clustering the signs into S= 3 sub-gestures. For each iteration of the algorithm, all time augmented observation vectors are clustered and then each cluster index sequence is analysed in order to determine if all clusters have a valid temporal clustering. The time scaling factor is increased and this process is repeated until all observation sequences have a valid temporal clustering. Figure 4.6 shows each step of the clustering algorithm when applied to the nine samples of the left hand data for the sign "Alot", while Figure 4.7 shows the results of the clustering for both the left and right hands.

The final temporal clusters are then used to divide the observation sequences into the S sub-gestures and the mean vector μ_j and the covariance

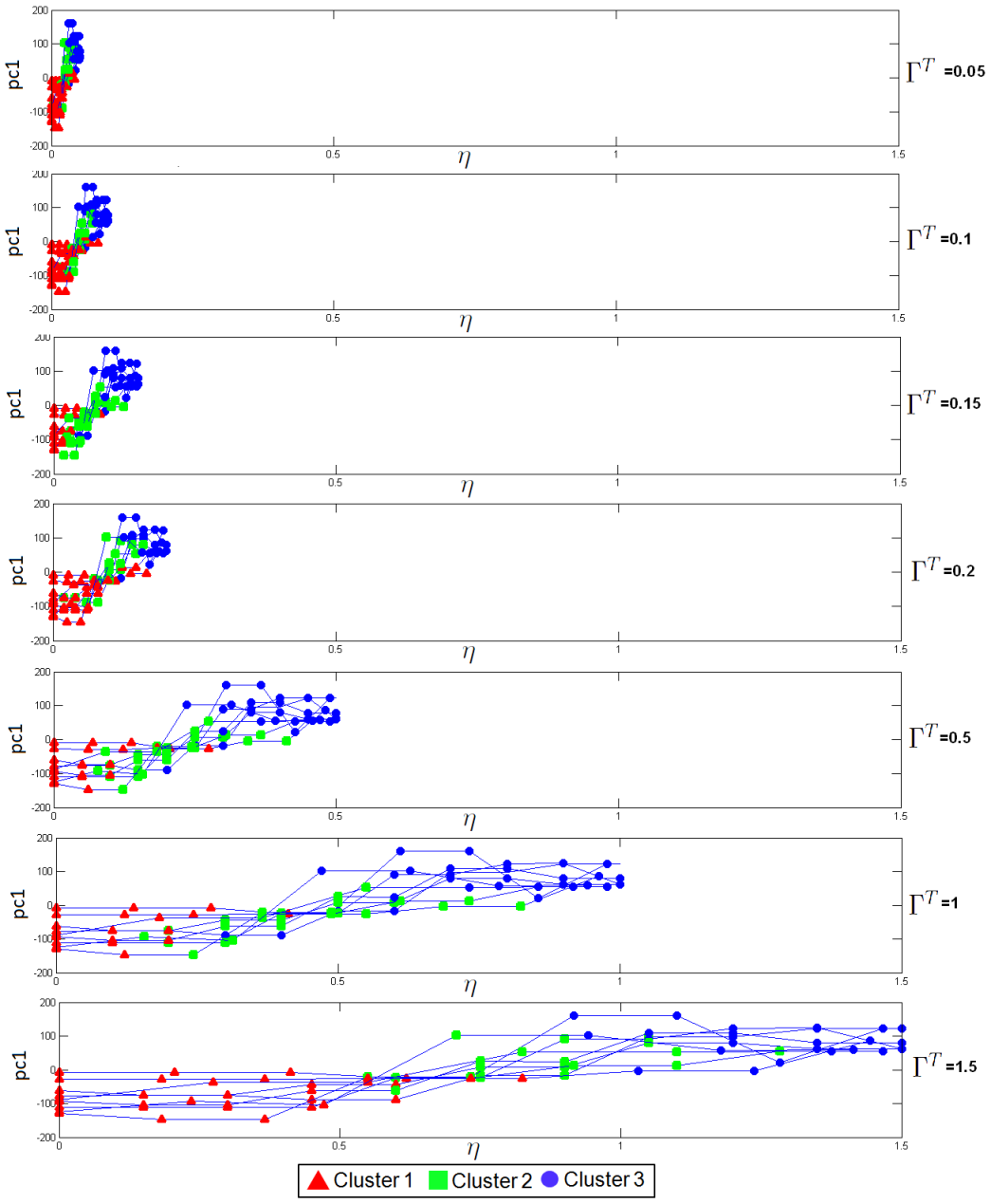


Figure 4.6: Iterative clustering steps. Each plot represents a clustering step with a different time scaling factor Γ^T. (pc1 - Principal Component)

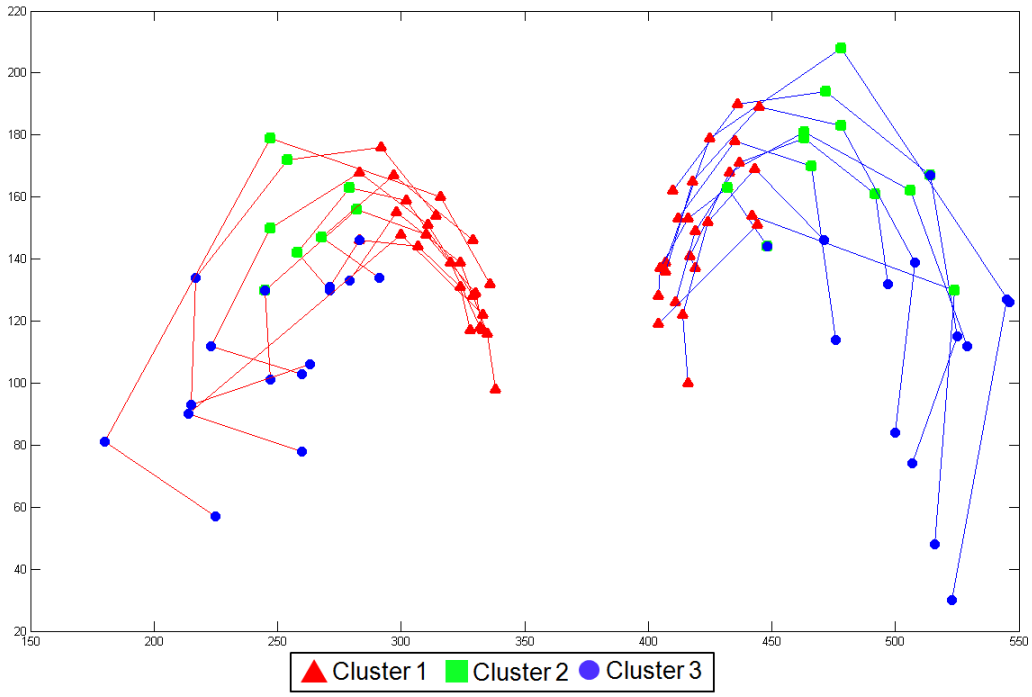


Figure 4.7: Temporal clustering results after final iteration

matrix Σ_j is calculated for each state (see Figure 4.8).

The Baum-Welch algorithm [Rab89] is then applied to λ_y using all training data Δ_y. After training, the Viterbi algorithm[Rab89] is run on each of the training sequences in Δ_y to produce the most probable state sequences. The initial S sub-gestures are then realigned to match the Viterbi paths. This re-estimation and realignment process is continued until the likelihood, produced by the Baum-Welch algorithm, converges. The overall process is repeated for different combinations of (S,R) to find the combination which produces the highest likelihood from the Baum-Welch re-estimation. Figure 4.9 gives an overview of the iterative training and parameter selection

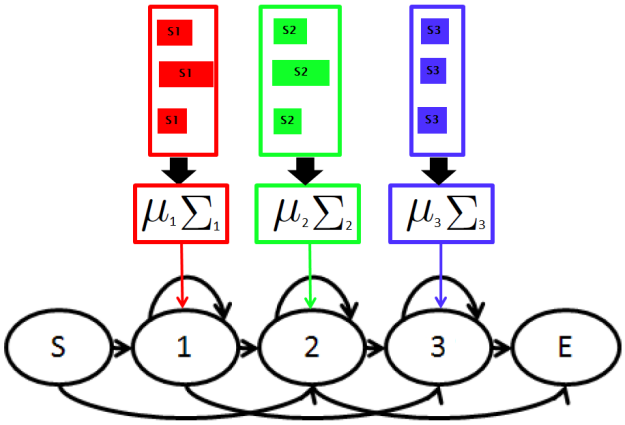


Figure 4.8: Initialisation: Mean vector μ and the covariance matrix Σ calculated for each HMM state

procedure.

After training, and finding the optimal parameters for each HMM λ_y, a threshold model $\overline{\lambda}$ is created using the method discussed in Section 4.3. Using each λ_y, and its corresponding set of states $S_y = \{s_{y1}, ..., s_{yN^y}\}$, the threshold model states $\overline{S}$ are initialised by copying all the HMM states such that $\overline{S} = \{s_{11}, ..., s_{1N^1}, ..., s_{y1}, ..., s_{yN^y}, ..., s_{Y1}, ..., s_{YN^Y}\}$, where N^y defines the number of states in λ_y. The set of HMMs, to recognise the Y pre-trained gestures, is then denoted as $\Lambda = \{\lambda_1, \lambda_2, ..., \lambda_Y, \overline{\lambda}\}$.

4.4.2 GT-HMM for Gesture Recognition

Gesture Classification

Given a sequence of gesture observations G, representing an unknown gesture, the goal is to accurately classify the gesture as an epenthesis or as one

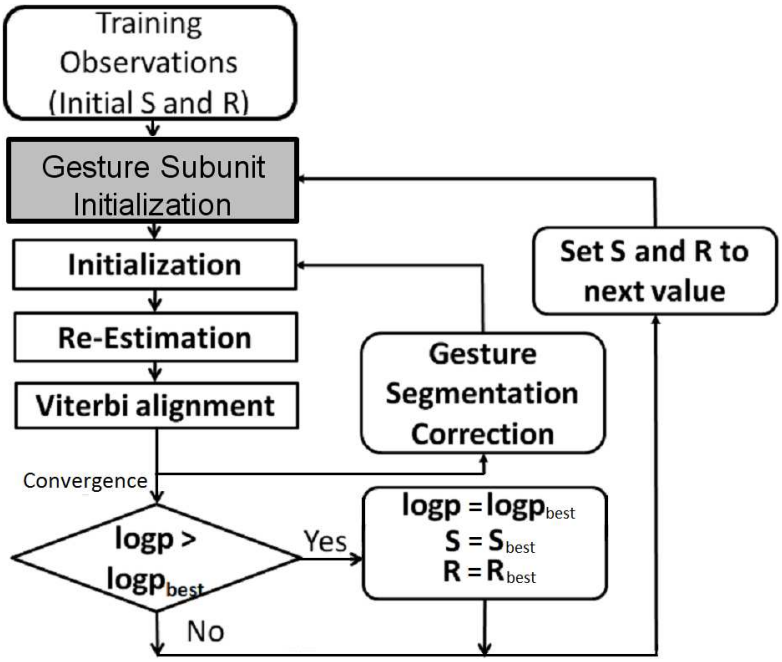


Figure 4.9: HMM Initialisation and Training Procedure

of the Y trained gestures. To classify the observations, the Viterbi algorithm is run on each model given the unknown observation sequences G, calculating the most likely state paths through each model y. The likelihoods of each state path, which we denote as $P(G|\lambda_y)$, are also calculated. The sequence of observations can then be classified as y if $P_{ML}(G|\lambda_y) \geq \tau_y$, where the maximum likelihood, $P_{ML}(G|\lambda_y)$, is defined in Equation 4.4 and the movement epenthesis likelihood, τ_y, is defined in Equation 4.5.

$$P_{ML}(G|\lambda_y) = \max_y P(G|\lambda_y) \tag{4.4}$$

$$\tau_y = P(G|\overline{\lambda})\Gamma_y \tag{4.5}$$

Where Γ_y is a constant scalar value used to tune the sensitivity of the system to movement epenthesis gestures.

Parallel Training

Vogler et al. [VM99, VA01] show that parallel HMMs can improve recognition rates of two-handed gestures when compared to standard HMMs. In order to recognise two-handed gestures in sign language, we implement a parallel GT-HMM system. The parallel GT-HMM initialises and trains a dedicated parallel HMM denoted as $\lambda_y^{'} = \{\lambda_{Ly},\lambda_{Ry}\}$ where λ_{Ly} and λ_{Ry} are HMMs which model the left and right hand gestures respectively. Each parallel GT-HMM is trained using the same gesture subunit initialisation and training technique discussed in Section 4.4.1.

To account for variations in information held in each of the hands for a particular sign, a weighting of ω_{Ly} and ω_{Ry} is applied to the left hand HMM and right hand HMM respectively. The weights are implemented to give more emphasis to the hand which conveys most information. For example if a signer's dominant hand performs a waving gesture while the non-dominant hand does not move, then more emphasis should be put on the dominant hand during the classification process. The weighting applied in our system is based on a hand variation ratio which calculates the relative variance between the left and right hand observation sequences. The weights are automatically calculated using training data from all observation sequences

G^k_{Ly} and G^k_{Ry}, where $1 \leq k \leq K$, K is the total number of training examples and G_{Ly} and G_{Ry} are the left and right hand observations respectively. The variance of the left and right hand observations are computed by calculating the variance of each observation dimension $\sigma^2_{Ly}[i]$ and $\sigma^2_{Ry}[i]$, where $0 \leq i \leq M$ and M is the dimension of the observation vectors. The left HMM weight, ω_{Ly}, and right HMM weight, ω_{Ry}, are then calculated using Equations 4.6 and 4.7, where $\omega_{Ry}+\omega_{Ly}= 1$.

$$\omega_{Ly} = \sum_{i=0}^{M} \frac{\sigma^2_{Ly}[i]}{(\sigma^2_{Ly}[i] + \sigma^2_{Ry}[i]) \times M} \tag{4.6}$$

$$\omega_{Ry} = \sum_{i=0}^{M} \frac{\sigma^2_{Ry}[i]}{(\sigma^2_{Ly}[i] + \sigma^2_{Ry}[i]) \times M} \tag{4.7}$$

A parallel GT-HMM framework, $\overline{\lambda}' = \{\overline{\lambda}_L,\overline{\lambda}_R\}$ is then created using the network of trained parallel HMMs λ_y $(y \in Y)$.

Parallel Gesture Classification

To classify the parallel observations $G' = \{G_L,G_R\}$, the Viterbi algorithm is run on each model given the unknown observation sequences G_L and G_R, calculating the most likely state paths through each model y. The likelihoods of each state path, which we denote as $P(G_L|\lambda_{Ly})$ and $P(G_R|\lambda_{Ry})$, are also calculated. We calculate the overall likelihoods of a two-handed gesture by computing the weighted sum of the left and right HMM likelihoods as defined in Equation 4.8.

$$P(G^{'}|\lambda_{y}^{'}) = P(G_L|\lambda_{Ly})\omega_{Ly} + P(G_R|\lambda_{Ry})\omega_{Ry} \tag{4.8}$$

The movement epenthesis likelihood is similarly calculated from a weighted sum of left and right threshold model likelihoods as defined in Equation 4.9.

$$\tau_{y}^{'} = \frac{P(G_L|\overline{\lambda_L})\Gamma_{Ly} + P(G_R|\overline{\lambda_R})\Gamma_{Ry}}{2} \tag{4.9}$$

Where Γ_{Ly} and Γ_{Ry} are constant scalar values used to tune the sensitivity of the system to movement epenthesis. In experiments discussed in Section 4.5, different scalar values are evaluated and results show that scaler values between 1.05 and 1.1 perform best when identifying epenthesis.

The observation sequence can then be classified as y if $P_{ML}(G^{'}|\lambda_{y}^{'}) \geq {\tau_y}^{'}$, where $P_{ML}(G^{'}|\lambda_{y}^{'})$ is the maximum likelihood defined as $\max_{y} P(G^{'}|\lambda_{y}^{'})$.

Continuous Recognition

Thus far we have described methods for classifying an isolated observation sequence as one of a number of pre-trained gestures or as a movement epenthesis. In this section we describe our system for spotting and classifying spatiotemporal gestures within continuous sequences of natural sign language.

The first step in our GT-HMM gesture spotting algorithm is gesture end point detection. To detect a gesture end point in a continuous stream of gesture observations $G = \{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_T\}$, we calculate the model likelihoods of observation sequence $G^* = \{\mathbf{f}_{T-L}, \mathbf{f}_{T-L-1}, ..., \mathbf{f}_T\}$ where G^* is a subset of G and L defines the length of the observation subset used. We set L to the

average length of the observation sequences used to train the system.

A candidate gesture, κ, with end point, $\kappa_e = T$, is flagged when $\exists y : P(G^*|\lambda_y) \geq \tau_y$. Figure 4.10 illustrates the likelihood time evolution of the hand gesture model "Lost" when given an observation sequence where the signer performs the "Lost" sign. It can be seen from Figure 4.10 that a number of candidate end points occur between $T = 16$ and $T = 21$.

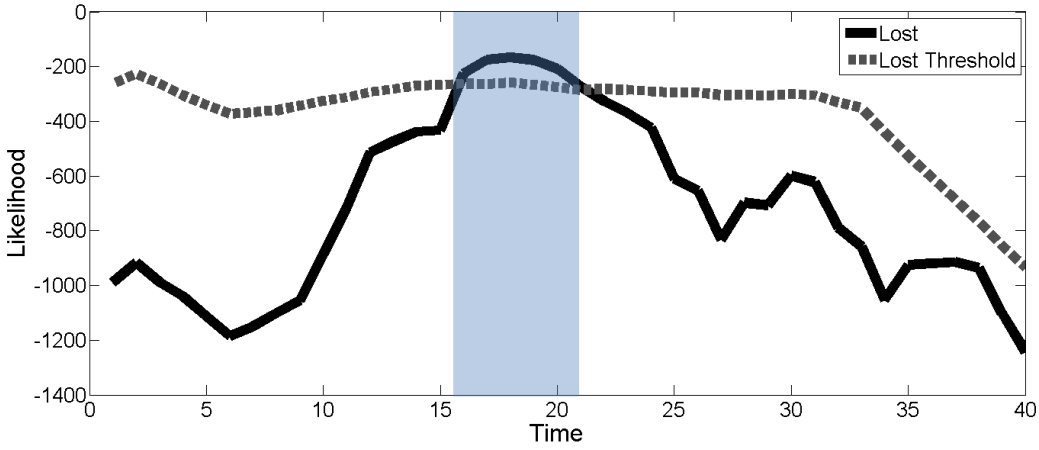


Figure 4.10: Likelihood evolution of "Lost" gesture model and associated threshold model

For each candidate end point, we calculate a corresponding start point κ_s. Different candidate start points are evaluated using the measurement shown in Equation 4.10 where $\beta_y(G)$ is a normalised metric ($0 \leq \beta_y(G) \leq 1$) which measures the likelihood of gesture y relative to the epenthesis likelihood given observations G.

$$\beta_y(G) = \frac{P(G|\lambda_y)}{P(G|\lambda_y) + \tau_y} \tag{4.10}$$

To find a candidate start point, the metric $\beta_y(G_{s\kappa_e})$ is calculated over different values of s, where $G_{s\kappa_e} = \{\mathbf{f}_s, \mathbf{f}_{s+1}, ..., \mathbf{f}_{\kappa_e}\}$ and $(\kappa_e - (L \times 2)) \leq s < \kappa_e$. The candidate gesture start point κ_s, is then found using Equation 4.11.

$$\kappa_s = \underset{s}{\text{argmax}} \beta_y(G_{s\kappa_e}) \tag{4.11}$$

Candidate Selection

The start and end point detection algorithm may flag candidate gestures which overlap and for this reason we expand on the continuous sign recognition algorithm with a candidate selection algorithm. The purpose of the candidate selection algorithm is to remove overlapping candidate gestures such that the single most likely gesture is the only remaining gesture for a particular time frame.

We use a sample sign language sentence "I Lost Alot of Books" to illustrate our candidate selection algorithm in the context of our gesture and threshold likelihood evaluation, where the system was trained on the following 8 signs: "Paper", "Alot", "Bike", "Clean", "Paint", "Plate", "Lost" and "Gone". Figure 4.11 illustrates the difference between the HMM gesture model likelihood $P(G|\lambda_y)$ and its corresponding threshold τ_y, where positive values indicate $P(G|\lambda_y) \geq \tau_y$. In Figure 4.11, four gesture model likelihoods are shown as all other gesture model likelihoods never exceed their corresponding threshold.

The first step in the candidate selection algorithm is to cluster overlapping gestures with the same gesture classification together. Each of these

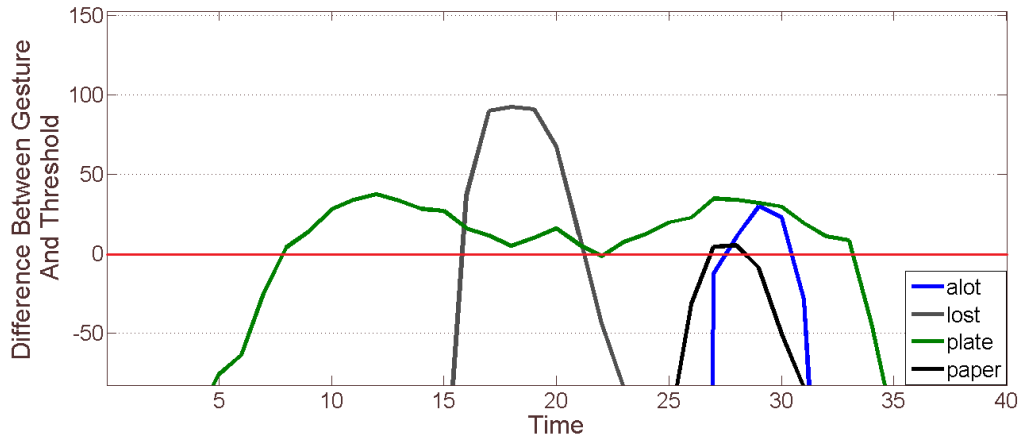


Figure 4.11: HMM Gesture Models And Corresponding HMM Threshold Model Likelihood Difference

candidate gestures within the cluster have an associated metric which we denote as κ_p=$\beta_y(G_{\kappa_s\kappa_e})$. We remove all but one candidate gesture from this cluster leaving only the candidate gesture, κ^B, with the highest κ_p value. We repeat this step for each cluster to produce a set of candidate gestures $\Upsilon = \{\kappa^{B1},\kappa^{B2},...,\kappa^{BK}\}$, where K is the total number of clusters created from grouping overlapping gestures, with the same gesture classification, together. Figure 4.12 shows the time segments and κ_p metrics of each candidate gesture after the first candidate selection step.

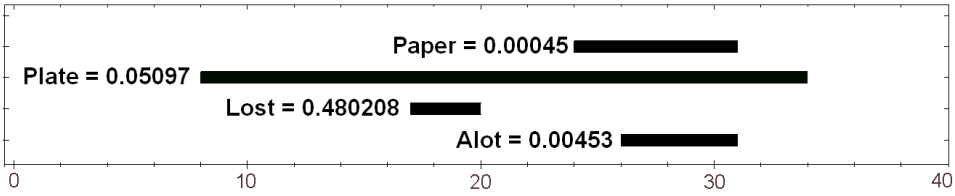


Figure 4.12: Candidate Gestures, Υ, after first candidate selection step

The second candidate selection step finds sets of overlapping candidates and removes the least probable candidates such that a maximum of only one candidate is detected for any given time frame. Figure 4.13 shows the time segments and gesture probabilities of the recognised gestures after the first and second candidate selection step where the signs "Lost" and "Alot" are correctly recognised from a sample sign language sentence "I Lost Alot of Books".

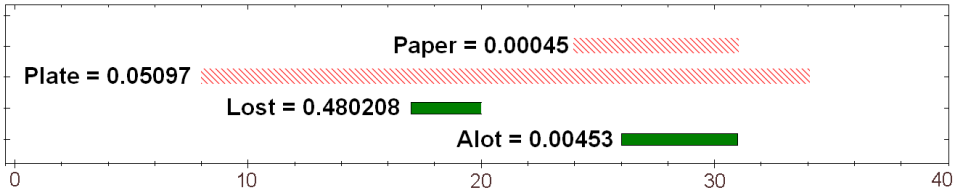


Figure 4.13: Candidate Gestures, Υ. Candidates marked in Red (Dashed) denote gestures which are removed by the second candidate selection step. Candidates in Green (Solid) denote the final recognised gestures

4.5 Experiments

In this section we evaluate our GT-HMM framework on a number of different gesture data sets and compare with different temporal event modeling frameworks including CRFs. Prior to discussing the experiments, we give a brief overview of CRFs, and extensions to CRFs, which we compare our GT-HMM framework to.

4.5.1 Conditional Random Fields

HMMs are generative models, assigning a joint probability to pairs of observations and labels. HMM parameters are typically trained to maximise the joint likelihood of training examples. To define a joint probability over observation and label sequences, a generative model needs to enumerate all possible observation sequences. HMMs typically require features appropriate for the particular recognition task and it is not practical to use feature vectors which are comprised of multiple interacting features. The main weakness of HMMs is the assumption of independence: the assumption that current observations are statistically independent of the previous observations. This is one of the main motivations for the use of CRFs. CRFs use an exponential distribution to model the entire sequence given the observation sequence. This avoids the assumption of independence between observations, and allows non-local dependencies between states and observations.

CRFs are a framework based on conditional probability approaches for segmenting and labeling sequential data. A CRF is an undirected graphical model in which each vertex represents a random variable whose distribution is to be inferred and each edge represents a dependency between two random variables. The task is to learn a mapping of observations $x \in X$ to class labels $y \in Y$, where x is a m dimensional vector of local observations, $x = \{x_1, x_2, ..., x_m\}$, and each local observation x_j is represented by a feature vector $\phi(x_j) \in \Re^d$. A conditional model $p(y|x)$ is constructed from the paired observation and label sequences.

A graph $O = (V, E)$ is defined such that $Y = (Y_v)_{v \in V}$, where V is the

set of vertices and E is the set of edges in the graph O. Then (X, Y) is a conditional random field, when conditioned on X, the random variables Y_v obey the Markov property with respect to the graph: $p(Y_v|X, Y_w, w \sim v)$ where $w \sim v$ implies that w and v are neighbours.

If $O = (V, E)$ of Y is a tree (a chain being the simplest form of a tree where $O = (V = \{1, 2, ...m\}, E = \{(i, i+1)\})$), then the joint distribution over label sequence Y given X is denoted in Equation 4.12.

$$p(y|x, \theta') \propto \exp\left(\sum_{e \in E,k} \lambda_k f_k(e, y|_e, x) + \sum_{v \in V,k} \mu_k g_k(v, y|_v, x)\right) \tag{4.12}$$

Where $y|_s$ is the set of components of y associated with the vertices of subgraph S, and feature functions f_k and g_k are assumed to be given and fixed. The parameter estimation problem is to determine the parameters $\theta' = (\lambda_1, \lambda_2, ...; \mu_1, \mu_2, ...)$ from training data $D = \{(x^{(i)}, y^{(i)})\}_i^N$ with empirical distribution $p^{\sim}(x, y)$.

Hidden Conditional Random Fields

Wang et al. [WQM+06] proposed a discriminative hidden-state approach for the recognition of human gestures. For any set of observations x they implement a set of hidden variables $s = \{s_1, s_2, ..., s_m\}$ which are not observed on training examples. Each s_j is a member of S where S is a finite set of possible parts in the model. Each s_j corresponds to a labeling of x_j with some member of S. A HCRF models the conditional probability of a class label given a set of observations by:

$$P(y|x,\theta') = \sum_s P(y,s|x,\theta') = \frac{\sum_s e^{\Psi(y,s,x;\theta')}}{\sum_{y' \in Y, s \in S^m} e^{\Psi(y',s,x;\theta')}} \tag{4.13}$$

The potential function $\Psi(y,s,x;\theta') \in \Re$, parameterised by θ', measures the compatibility between a label, a set of observations and a configuration of the hidden states.

Latent-Dynamic Conditional Random Fields

The CRF approach models the transitions between gestures, thus capturing extrinsic dynamics, but lacks the ability to represent internal sub-structure. Each Hidden-state Conditional Random Field models a single gesture label but cannot learn the dynamics between gesture labels. Morency et al. [MQD07] propose a LDCRF to combine the strengths of CRFs and HCRFs by capturing both extrinsic dynamics and intrinsic sub-structure. They define the latent conditional model as shown in Equations 4.14 - 4.17.

$$P(y|x,\theta') = \sum_s P(y|s,x,\theta')P(s|x,\theta') \tag{4.14}$$

$$P(s|x,\theta') = \frac{1}{Z(x,\theta')} \exp(\sum_k \theta'_k \cdot F_k(s,x)) \tag{4.15}$$

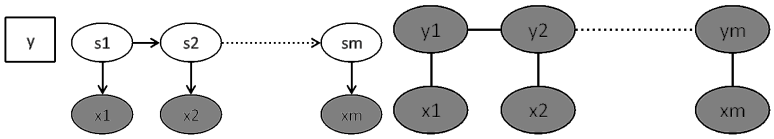
$$Z(x,\theta') = \sum_s \exp(\sum_k \theta'_k \cdot F_k(s,x)) \tag{4.16}$$

Where F_k is defined as

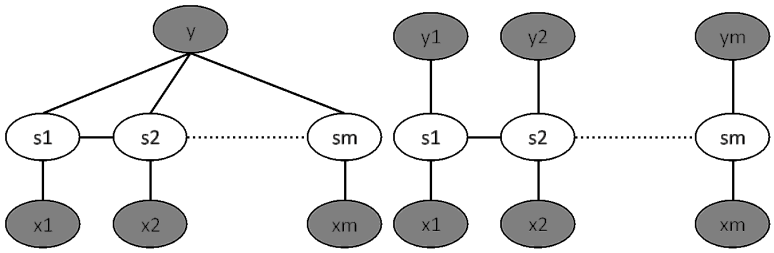
$$F_k(s, x) = \sum_{j=1}^{m} f_k(s_{j-1}, s_j, x, j) \tag{4.17}$$

Each feature function f_k is either a state function $s_k(s_j, x, j)$ or a transition function $t_k(s_{j-1}, s_j, x, j)$.

Figure 4.14 illustrates the graphical models of HMM, CRF, HCRF and LDCRF.



(a) HMM (b) CRF



(c) HCRF (d) LDCRF

Figure 4.14: Comparison of HMM and different CRF models where grey circles denoted observed symbols.

CRFs For Gesture Recognition

Wang et al. [WQM+06] and Morency et al. [MQD07] propose a gesture recognition framework using HCRFs and LDCRFs respectively. We evaluate

this same framework for the recognition of motion based gestures in sign language.

CRF Training Similar to the works of Wang et al. and Morency et al., we implement an objective function, shown in Equation 4.18, to train the parameters of each of the CRF models.

$$L(\theta') = \sum_{i=1}^{n} \log P(y_i|x_i, \theta') - \frac{1}{2\sigma^2}\|\theta'\|^2 \tag{4.18}$$

Where n is the total number of training sequences. We implement a gradient ascent search to find the optimal parameter values, $\theta^* = \underset{\theta'}{\text{argmax}} L(\theta')$ using a Quasi-Newton optimisation technique.

CRF Gesture Classification Given an unknown sequence of gesture observations G, we calculate the conditional probability $P(y|G, \theta')$ of each of the CRF, HCRF and LDCRF models for gesture labels $y \in Y$.

We classify a given observation sequence G as gesture class y if $P_{ML}(y|G, \theta') > \Omega$, where Ω is a pre-defined threshold value and $P_{ML}(y|G, \theta')$ is the maximum likelihood defined as $\max_{y} P(y|G, \theta')$.

CRF Parallel Training Similar to the parallel GT-HMM system, we implement a parallel CRF model in order to recognise two-handed spatiotemporal gestures. We apply the same weighting technique, discussed in Section 4.4.2, to the parallel CRF models by calculating left hand CRF weights, ω_{Ly} and right hand CRF weights ω_{Ry}.

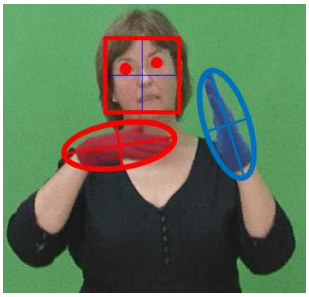


Figure 4.15: Extracted Features from Image

CRF Parallel Classification Given a parallel observation sequence $G^{'} = \{G_L, G_R\}$, we calculate the conditional probability $P(y|G_L, \theta)$ and $P(y|G_R, \theta)$ for each parallel CRF model. The parallel conditional probability is then defined in Equation 4.19.

$$P(y|G^{'}, \theta') = P(y|G_L, \theta')\omega_{Ly} + P(y|G_R, \theta')\omega_{Ry} \tag{4.19}$$

We classify a given observation sequence $G^{'}$ as gesture class y if $P_{ML}(y|G^{'}, \theta') > \Omega^{'}$, where $\Omega^{'}$ is a pre-defined threshold value and $P_{ML}(y|G^{'}, \theta')$ is the maximum likelihood defined as $\max_{y} P(y|G^{'}, \theta')$.

4.5.2 Feature Extraction

In this section we conduct evaluations on different gesture recognition systems using data extracted from video sequences of sign language sentences being performed by a fluent ISL signer.

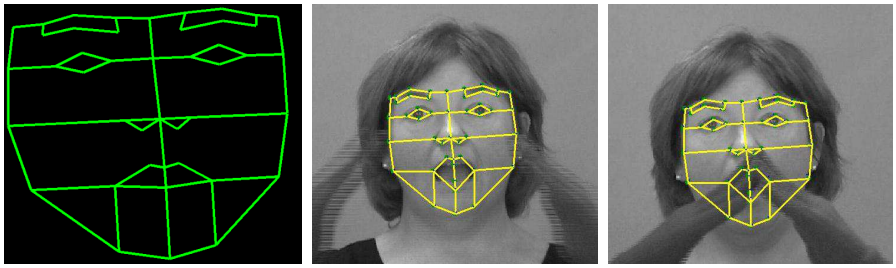
For completeness, prior to discussing the experiments, we briefly describe the feature extraction techniques implemented.

Tracking of the hands is performed by tracking coloured gloves using the Mean Shift algorithm [CRM00]. Face and eye positions are used as features for head movement recognition and also used as hand gesture cues. Face and eye detection is carried out using a cascade of boosted classifiers working with haar-like features proposed by Viola and Jones [VJ01]. A set of public domain classifiers [MCSLN08], for the face, left eye and right eye, are used in conjunction with the OpenCV implementation of the haar cascade object detection algorithm. Figure 4.15 shows a visual example of the features of an ISL signer being tracked.

We define the raw features extracted from each image as follows; right hand position (RH_x, RH_y), left hand position (LH_x, LH_y), face position (FC_x, FC_y), face width (FW) (face region is square),left eye position (LE_x, LE_y) and right eye position (RE_x, RE_y).

In order to recognise non-manual signals conveyed through facial expressions, we locate the facial features of interest using Cootes' implementation of Active Shape Models (ASM) [CT01]. ASMs are statistical models which can be used to carry out facial feature localisation. The ASM is trained to deform iteratively to fit new facial images. For the experiments conducted as part of this work, our training set consisted of 3500 images in total. From which 300 key frames representing the variance in the data set were manually labeled with 46 points. Figure 4.16(a) shows the ASM which was trained on these image-points pairs.

During sign language communication, the face is frequently occluded by the hands. To overcome this, we fit the ASM to the parts of the face that are



(a) (b) (c)

Figure 4.16: (a) sample ASM which was fitted to each image (b) sample of an un-occluded image (c) example of an occluded image

visible and use previous points for occluded parts of the face. This can be seen in Figure 4.16 where the position of the mouth from Figure 4.16(b) is used in Figure 4.16(c) when the mouth is occluded. This is a valid approach as the hands move rapidly and rarely cover the same portion of the face for multiple frames.

4.5.3 Evaluation of Techniques on Isolated Gestures

Wang et al. [WQM+06] performed experiments to show that the HCRF model performed better at classifying head and arm gestures than CRFs and HMMs. In their experiments, the models were evaluated on their ability to classify a given segmented gesture sequence as one of a number of pre-trained gestures but the models were not tested on non-gesture sequences. In order to evaluate and assess the ability of a HCRF model to recognise gestures in sign language, the performance of the model must be evaluated when identifying non-gestures and epenthesis as well as being evaluated on the performance

of classifying gestures.

Morency et al. [MQD07] performed experiments to evaluate the performance of the LDCRF model on three different data sets. The first data set was a head nod data set where the system was trained and tested on frames labeled as a head nod or labeled as not a head nod. The second data set, similar to the first data set, was trained and tested on positive and negative examples of heads nods. The final data set was an eye gaze data set, and the system was trained and tested on frames labeled as either an eye gaze-aversion gesture or a non gaze-aversion gesture. The LDCRF model was shown to out-perform CRF, HCRF and HMM based classifiers (as well as a support vector machine based classifier). From these experiments it is difficult to access whether the LDCRF model could be implemented to recognise a larger vocabulary of gestures or whether the LDCRF model could be used in a sign language based system. In the experiment Morency et al. carry out, each of the gesture data set experiments were trained to recognise a single gesture with positive and negative examples of the gesture. In order to evaluate the LDCRF model for a sign language recognition system, the model should be tested on a larger vocabulary of gestures. In their experiments the gesture model was trained on positive and negative examples of the gesture. Training a model to recognise movement epenthesis in sign language is unfeasible due to the large number of possible epenthesis that can occur between signs.

The goal of the experiments conducted in this section is to evaluate the performance of the HMM, Threshold Hidden Markov Model (T-HMM),

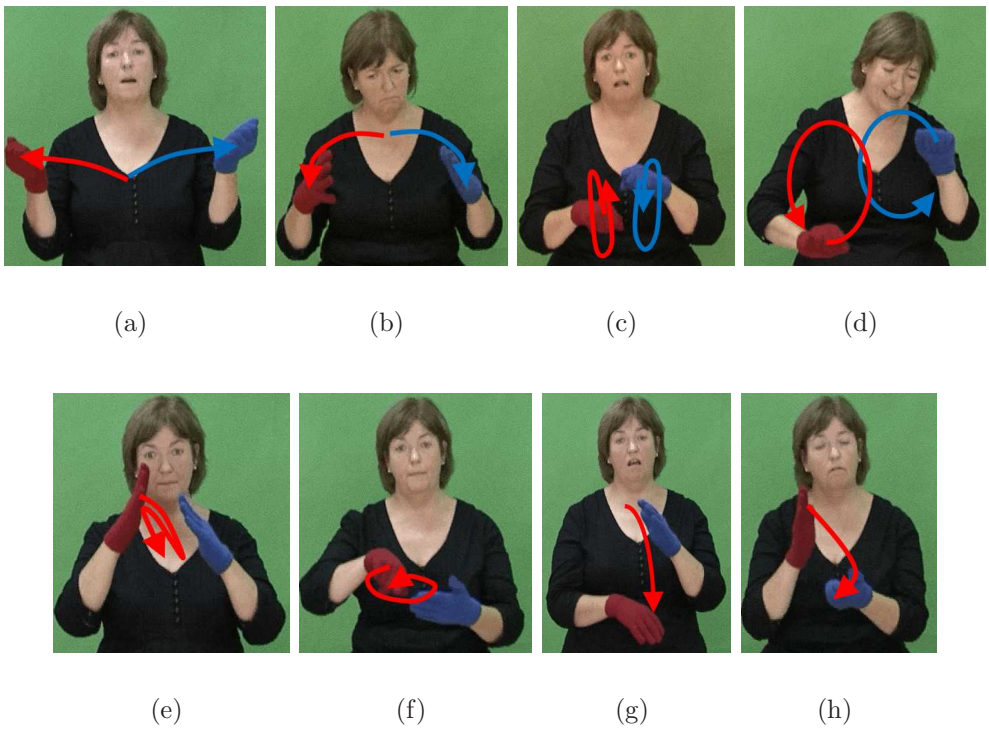
GT-HMM and the different CRF models when recognising motion based gestures and identifying epenthesis which occur in sign language. The T-HMM model we evaluate in this work is a parallel threshold HMM framework where training and classification are carried out in the same manner as the GT-HMM model. The key difference between the GT-HMM and the T-HMM is that the T-HMM is not trained using the gesture subunit initialisation technique described in Section 4.4.1. Instead, the T-HMM model is initialised using a standard segmentation method, utilised by Holden et al. [HR01], where the observation sequences are linearly segmented into S equal sub-sequences.

Since sign language communication is multimodal it involves not only hand gestures (i.e. manual signing) but also non-manual signals conveyed through facial expressions, head movements, body postures and torso movements [COR05]. In order to evaluate the use of HMMs and CRFs in recognising motion based gestures in sign language, we evaluate the models on three data sets; a manual signing data set (i.e. two-handed motion based gestures) and two non-manual signal data sets based on head motion gestures and eye brow gestures.

Manual Sign Experiments

The first data set we use to evaluate the models is a set of two-handed spatiotemporal hand gestures used in sign language. This data set consists of eight different manual signs extracted from videos of a fluent signer performing natural sign language sentences. Figure 4.17 illustrates an example of a

signer performing each of the eight manual signs.



(a) (b) (c) (d)

(e) (f) (g) (h)

Figure 4.17: Example of the eight different signs the system was tested on (performed by Signer 1): (a) Newspaper, (b) A lot, (c) Bike, (d) Clean, (e) Paint, (f) Plate, (g) Lost, (h) Gone

In order to recognise manual signs, we must extract two observation channels from the video streams. These observation channels correspond to the left hand observations G_L and the right hand observations G_R. The observations G_L and G_R are combined into a parallel observation sequence G' which is processed by the parallel models. We extract a set of observation sequences $\Delta_y{}'$ from the video sequences, where $y \in Y$, Y is the set of sign labels, $\Delta_y{}' = \{G'_{1y}, ...G'_{Ty}\}$ and T is the number of sample observation se-

quences recorded for each gesture label y.

This set is then divided into a training set, ${\Delta_y}'^t$, and a test set, ${\Delta_y}'^\zeta$. A set of 10 training signs and a set of 10 test signs were recorded for each sign (A total of 160 gesture samples). The HMM, T-HMM, GT-HMM, CRF, HCRF and LDCRF models were then trained on ${\Delta_y}'^t$.

An additional set of observations Δ'_E, which represents a collection of movement epenthesis, were also extracted from the video sequences to test the performance of the threshold model. For each sign, 10 movement epenthesis that occurred before and after the sign in different sign language sentences were recorded. An additional set of 20 random movement epenthesis were also recorded, resulting in a test set of 100 samples to evaluate the models on.

To evaluate the performance of the models, we perform a ROC analysis on the different models and calculate the AUC for each model. The classification of a gesture is based on a comparison of a model probability and a threshold value. In our ROC analysis of each model, we vary the threshold and create a confusion matrix for each of the thresholds. In the case of the T-HMM and GT-HMM models, we vary the weighting of the threshold and in the case of the CRF models we vary the static threshold value Ω.

When evaluating the HCRF and LDCRF models, we test the models on different numbers of hidden states and different window parameters ω. The window parameter defines the amount of past and future history to be used when predicting the state at time t such that long range dependencies can be incorporated. In our experiments we test each model on two different groups

of data. The first data group, which we denote as data set 1, is a set which includes all test sequences ${\Delta_y}^{'\zeta}$ and epenthesis sequences $\Delta_E^{'}$. The second data group, which we denote as data set 2, is a set which includes just the test sequences ${\Delta_y}^{'\zeta}$.

While an extensive evaluation of the models using different feature vectors was conducted, we report the results of models using the best performing feature vectors. The best performing feature vector for the HMM models was the feature, $\mathbf{f}= \{RP_x, RP_y, V_x, V_y, D_H\}$, which describes the position of the hands relative to the eyes, the direction of the movement of the hand and the distance between the two hands. The best performing feature vector for the three different CRF models was the feature vector $\mathbf{f}= \{V_x, V_y\}$, which describes the direction of the movement of the hand.

Table 4.1 shows the AUC measurements of the HMM, T-HMM, GT-HMM and different variations of the CRF models when classifying gestures using their corresponding best performing feature vector.

Table 4.1: Manual Signs: AUC Measurements for Different Models

Model	**Data Set** $1^{\dagger}$	**Data Set** $2^{\ddagger}$
HMM	0.902	0.943
GT-HMM	0.976	0.977
T-HMM	0.941	0.944
CRF $\omega = 0$	0.833	0.876
CRF $\omega = 1$	0.794	0.828
HCRF $\omega = 0$, $S = 6$	0.909	0.944
HCRF $\omega = 1$, $S = 6$	0.957	0.983
HCRF $\omega = 2$, $S = 6$	0.944	0.971
HCRF $\omega = 0$, $S = 8$	0.947	0.965
HCRF $\omega = 1$, $S = 8$	0.934	0.968
LDCRF $\omega = 0$, $S^* = 1$	0.847	0.881
LDCRF $\omega = 0$, $S^* = 2$	0.806	0.842
LDCRF $\omega = 0$, $S^* = 3$	0.808	0.836
LDCRF $\omega = 0$, $S^* = 4$	0.863	0.901
LDCRF $\omega = 0$, $S^* = 8$	0.942	0.985
LDCRF $\omega = 1$, $S^* = 8$	0.899	0.928

† - Data Set which includes 100 epenthesis samples
‡ - Data Set which does not include epenthesis samples
$*$ - S^* refers to number of hidden states per label for LDCRF

These results show that the overall best performing model, with an AUC of 0.985, was the LDCRF model with 8 hidden states per label when tested on data set 2. However, a sign language recognition system must be able to identify movement epenthesis as well as classify gestures. The results of the tests performed on data set 1 are more representative of how the models would perform in real world sign language recognition scenarios. The model which scores best when classifying data set 1 is the GT-HMM which has an AUC of 0.976. Although the HCRF and LDCRF perform better than the GT-HMM when classifying gestures, the performance of both drop

significantly when the epenthesis data is introduced. The performance of the GT-HMM drops by 0.1% compared to the relatively large drop of 4.3% by the LDCRF model. The significance of this result is that it shows that our GT-HMM model can robustly identify movement epenthesis and classify gestures. While the CRF models perform well at classifying gestures, they perform poorly at identifying movement epenthesis.

The results of the experiments also reveal the influence that our gesture subunit initialisation has on the recognition performance. The GT-HMM framework initialises the HMM using our initialisation method described in Section 4.4.1, while the T-HMM utilises a linear segmentation method. The experiments show that the GT-HMM results in an improvement of 3.5% when compared to the T-HMM model.

Head Gesture Experiments

The second data set we use to evaluate the HMM and CRF models is a set of head movement gestures. The head movement gestures are used to convey non-manual information in sign language sentences. The head gesture set consists of three different head movement gestures extracted from videos of a fluent signer performing natural sign language sentences.

A visual example of a signer performing each of the three different head movement gesture is shown in Figure 4.18.

Similar to the manual sign experiments described in Section 4.5.3, observation sequences $\Delta_y = \{G_{1y}, ...G_{Ty}\}$ were extracted from the videos and divided into a training set, ${\Delta_y}^t$, and a test set, ${\Delta_y}^\zeta$. For the head movement

(a)

(b)

(c)

Figure 4.18: Example of the three different head movement gestures the system was tested on (a) Right Movement (b) Left Movement (c) Left Forward Movement

experiments, a set of 6 training signs and a set of 6 test signs were recorded for each sign (A total of 36 gesture samples). The HMM models and all CRF models were then trained on ${\Delta_y}^t$.

A set of 25 additional head gesture sequences Δ_E, outside of the training set, were also extracted from the video sequences to test the performance of the system when identifying movement epenthesis. Similar to the hand gesture experiments, we test the head gesture models on two data groups; data set 1 includes the gesture test sequences and the epenthesis sequences, while data set 2 includes only the gesture test sequences.

While a ROC analysis of the non-manual models is conducted using the same procedure described in Section 4.5.3, we report the results of models using the best performing feature vectors. The best performing feature vector for the HMM models, when classifying head gestures, is a 2 dimensional vector $\mathbf{f}= \{V_x, V_y\}$ describing the velocity of the head movement in the x and y directions. To calculate the velocity vector of the head we use the mid point between the eyes and calculate the movement of the midpoint from frame to frame. As with the HMM models, the best performing feature vector for the CRF models is the 2 dimensional velocity vector $\mathbf{f}= \{V_x, V_y\}$. Table 4.2 shows the AUC measurements of the HMM, T-HMM, GT-HMM and different variations of the CRF models when classifying head movements using their corresponding best performing feature vector.

Table 4.2: Non-Manual Signals: AUC Measurements for Different Models

Model	**Data Set** 1†	**Data Set** 2‡
HMM	0.848	0.891
GT-HMM	0.936	0.947
T-HMM	0.873	0.882
CRF $\omega = 0$	0.736	0.768
CRF $\omega = 1$	0.527	0.545
HCRF $\omega = 0$, $S = 2$	0.698	0.801
HCRF $\omega = 1$, $S = 2$	0.786	0.911
HCRF $\omega = 2$, $S = 2$	0.702	0.816
HCRF $\omega = 0$, $S = 4$	0.784	0.927
HCRF $\omega = 1$, $S = 4$	0.719	0.811
HCRF $\omega = 0$, $S = 6$	0.743	0.850
HCRF $\omega = 1$, $S = 6$	0.736	0.893
HCRF $\omega = 0$, $S = 8$	0.715	0.838
HCRF $\omega = 1$, $S = 8$	0.708	0.788
LDCRF $\omega = 0$, $S^* = 3$	0.794	0.899
LDCRF $\omega = 1$, $S^* = 3$	0.763	0.880
LDCRF $\omega = 0$, $S^* = 6$	0.760	0.827
LDCRF $\omega = 1$, $S^* = 6$	0.717	0.791
LDCRF $\omega = 0$, $S^* = 9$	0.868	0.922
LDCRF $\omega = 1$, $S^* = 9$	0.837	0.901
LDCRF $\omega = 2$, $S^* = 9$	0.894	0.952
LDCRF $\omega = 3$, $S^* = 9$	0.795	0.861

† - Data Set which includes 25 non-gesture samples
‡ - Data Set which does not include non-gesture samples
$*$ - S^* refers to number of hidden states per label for LDCRF

The results of this experiment repeat the same trend found in the results of the manual sign recognition experiment. The LDCRF model performs best when classifying gestures in data set 2. The recognition rate of the CRF models then decrease significantly when epenthesis are introduced. The best performing model for data set 1 is again the GT-HMM model with an AUC

of 0.936. The GT-HMM achieved a 4.2% higher AUC than the LDCRF when tested on data set 1. This result further suggests that the GT-HMM model is a more robust model for recognising gestures when epenthesis gestures are taken in to account.

Similar to the results achieved on the manual gestures in Section 4.5.3, these results indicate that the gesture subunit initialisation of the GT-HMM improved classification performance. Experiments show that the GT-HMM results in an improvement of 6.3% when compared to the T-HMM which is initialised using a standard HMM segmentation method.

Eye Brow Gesture Experiments

The third data set we use to evaluate the HMM and CRF models is a set of eye brow movement gestures used to convey non-manual information in sign language. Research on ASL conducted by Grossman *et al.* [GK06] has linked eyebrow gestures to certain affective states and questions. Anger, wh-questions (who, where, what, when, why, how) and quizzical questions exhibited lowered brows and squinted eyes, while surprise and yes/no questions showed raised brows and widened eyes. In experiments conducted in this section, we focus on identifying these lowered brow gestures and raised brow gestures.

Eyebrow observation sequences $\Delta_y = \{G_{1y}, ...G_{Ty}\}$ are extracted from the videos and divided into a training set, ${\Delta_y}^t$, and a test set, ${\Delta_y}^\zeta$. For the eye brow experiments, a set of 5 training signs and a set of 5 test signs were recorded for each sign (A total of 20 gesture samples). The HMM models

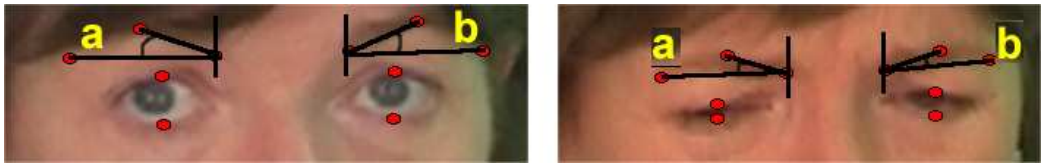


Figure 4.19: Example of subject performing a raised brow gestures (left) and a lowered brow gesture (right). a and b represent the angles ϕ_L and ϕ_R respectively

and all CRF models are then trained on ${\Delta_y}^t$.

An additional set of 20 other eye brow gesture sequences Δ_E, outside of the training set, are also extracted from the video sequences to test the performance of the system when identifying movement epenthesis.

The best performing feature vector for the HMM models and CRF models, when classifying eye brow movements, is a 2 dimensional vector $\mathbf{f}$= $\{\phi_{LR}, D^{\Delta}\}$. The value ϕ_{LR} is the average angle between ϕ_L and ϕ_R shown in Figure 4.19, and D^{Δ} is the change in distance between the two eyes.

We carry out a ROC analysis of the non-manual models using the same procedure as used in the hand gesture and head movement experiments. Table 4.3 shows the AUC measurements of the models.

Table 4.3: Non-Manual Eye Brow Signals: AUC Measurements for Different Models

Model	**Data Set** $1^{\dagger}$	**Data Set** $2^{\ddagger}$
HMM	0.882	0.911
GT-HMM	0.948	0.951
T-HMM	0.905	0.912
CRF $\omega = 0$	0.859	0.899
CRF $\omega = 1$	0.878	0.921
CRF $\omega = 3$	0.889	0.933
CRF $\omega = 6$	0.866	0.901
HCRF $\omega = 0$, $S = 1$	0.525	0.533
HCRF $\omega = 0$, $S = 3$	0.858	0.922
HCRF $\omega = 2$, $S = 3$	0.809	0.899
HCRF $\omega = 5$, $S = 3$	0.781	0.912
HCRF $\omega = 0$, $S = 6$	0.825	0.922
HCRF $\omega = 2$, $S = 6$	0.802	0.922
LDCRF $\omega = 0$, $S^* = 1$	0.788	0.891
LDCRF $\omega = 1$, $S^* = 1$	0.892	0.911
LDCRF $\omega = 5$, $S^* = 1$	0.913	0.944
LDCRF $\omega = 10$, $S^* = 1$	0.893	0.922
LDCRF $\omega = 0$, $S^* = 3$	0.888	0.932
LDCRF $\omega = 3$, $S^* = 3$	0.912	0.931
LDCRF $\omega = 5$, $S^* = 3$	0.918	0.955
LDCRF $\omega = 10$, $S^* = 3$	0.905	0.933
LDCRF $\omega = 0$, $S^* = 6$	0.864	0.896
LDCRF $\omega = 5$, $S^* = 6$	0.912	0.944
LDCRF $\omega = 10$, $S^* = 6$	0.890	0.912

† - Data Set which includes 20 non-gesture samples
‡ - Data Set which does not include non-gesture samples
$*$ - S^* refers to number of hidden states per label for LDCRF

The results of the eyebrow experiment repeat the same trend found in previous results of the hand gestures and head gestures. The LDCRF model performs best when classifying gestures in data set 2. The recognition rate of

the CRF models then decrease significantly when epenthesis are introduced. The best performing model for data set 1 is again the GT-HMM with an AUC of 0.948. The GT-HMM results in a 3% higher AUC than the LDCRF when tested on data set 1. This result further suggests that the GT-HMM model is a more robust model for recognising gestures when epenthesis gestures are taken in to account.

There was a 3.7% decrease in performance between the LDCRF from data set 2 to data set 1, while there was only a 0.3% decrease in performance of the HMM threshold model. This result suggests that the performance of the LDCRF would decrease more than that of the GT-HMM model when the number of epenthesis gestures introduced into the system increased.

Experiments also show that the GT-HMM model, initialised using our automated initialisation technique, had a 4.3% better AUC measurement than the T-HMM initialised using the standard HMM segmentation method.

Benchmark Data-set: Marcel InteractPlay Database

In this section we discuss a user independent experiment which was conducted to evaluate the performance of the models when recognising gestures performed by signers not represented in the training set. The user independent data set we utilise is a benchmark spatiotemporal hand gesture data-set. While this data-set does not include sign language gestures or movement epenthesis data, we utilise this data set in order to evaluate the performance of the GT-HMM framework when performing user independent gesture recognition and compare with additional gesture recognition techniques.

The database contains 3D hand trajectories of isolated hand gestures, in addition to 3D coordinates of the head and the torso. The database consists of 16 gestures carried out by 20 different people. For each person, there exists 50 samples of each gesture resulting in a total of 1000 gesture samples for each of the 16 gesture classes. 3D trajectories of the hands were obtained using a stereo vision system which tracks coloured gloves, a coloured t-shirt and the face in real-time using the EM algorithm (see Figure 4.20). A detailed description of the 3D blob tracking can be found in [BC01].

Just et al. [JM09] conduct a comparative study, using the InteractPlay data-set, to compare two state-of-the-art techniques for temporal event modeling. In their work, HMMs and IOHMMs (an extension to HMMs first proposed by Bengio and Frasconi [BF95]) are evaluated on the InteractPlay data-set. A 12 dimensional feature vector, comprising left and right hand positions and velocities, is used to describe gesture sequences. In order to improve user-independent recognition, each user performs a calibration pose and all other feature vectors are then normalised using the calibration pose. Training was carried out using all 8000 gesture sequences from 10 of the 20 people in the data-set. Testing was then carried out on 8000 gesture sequences from the remaining 10 people not used in training. Results of their experiments show that the HMM and Input/Output Hidden Markov Model (IOHMM) achieved an average recognition rate of 75% and 63% respectively.

We carry out a number of evaluation protocols when testing the GT-HMM framework on the Interactplay data-set:

- P1 - To investigate the influence the number of training subjects used

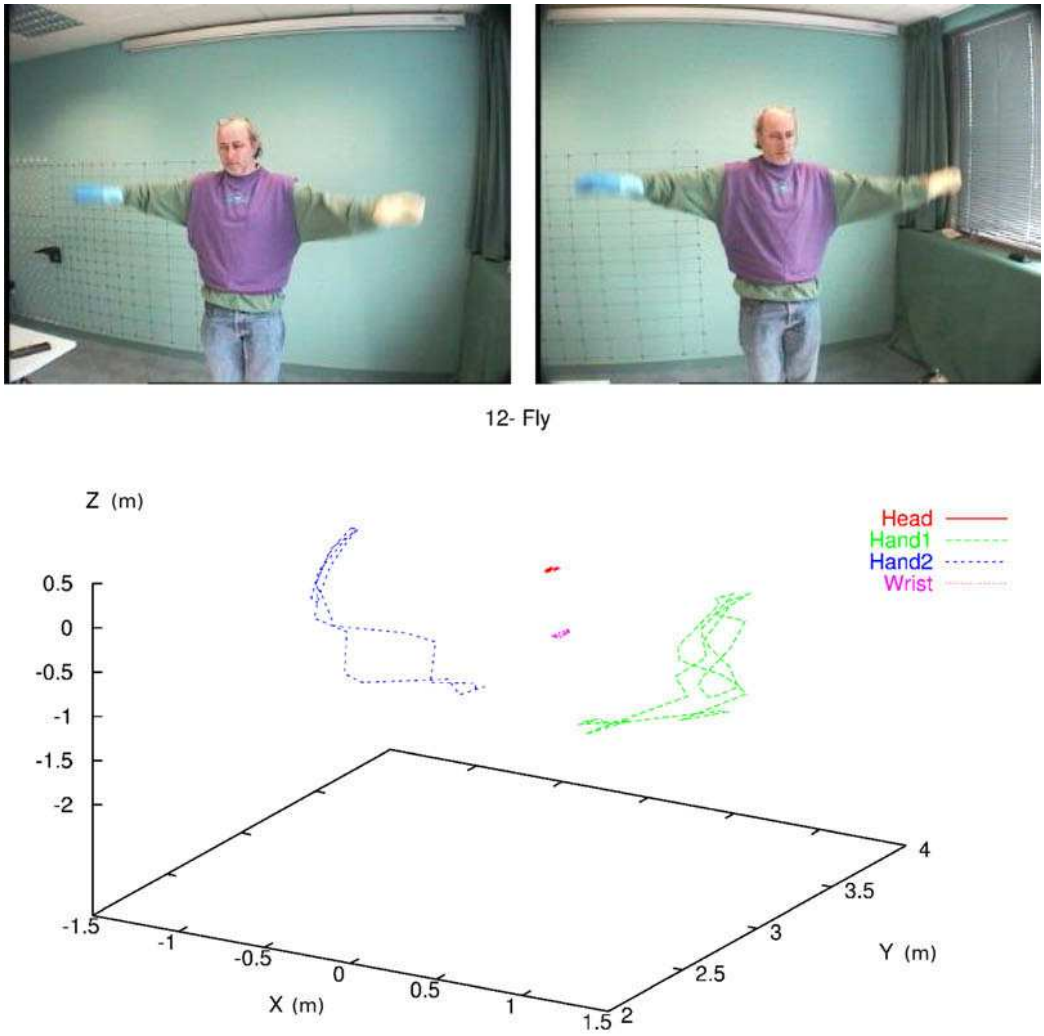


Figure 4.20: Top: example of images from a gesture sequence from the point of view of the left camera (on the left) and from the point of view of the right camera (on the right). Bottom: 3D coordinates of the centre of each blob (head, torso, left hand and right hand) for a fly gesture. The z-axis is the vertical axis of the person.

has on the overall recognition performance, we first train the gesture modeling frameworks using data from only 1 subject. This training set consisted of 50 samples of each gesture class. We then test the models on gesture sequences obtained from 10 subjects. Feature vectors are comprised of 3D hand positions and velocities.

- P2 - In order to evaluate the different gesture modeling frameworks, and directly compare results with the work of Just et al., we use the same training and testing protocol as described in their work. This consists of training the framework using data from 10 subjects and testing the framework on the remaining 10 subjects.

One of the overall goals of this work is to create a gesture recognition framework with minimal supervision during training. Just et al. utilise calibration poses to normalise different user inputs and improve recognition of user independent gestures. The process of creating a set of calibration poses for each new user requires further supervision and thus all evaluations of our GT-HMM framework are carried out without using any calibration data.

Table 4.4 details the results achieved by the models implemented by Just et al. and the results achieved by the LDCRF, T-HMM and GT-HMM frameworks when tested on the different evaluation protocols. Isolated gesture recognition results reported in this Chapter thus far have been presented using the ROC AUC measurement. While we present both ROC AUC and accuracy measurements in Table 4.4, in this Section we discuss accuracy measurement results in order to directly compare performance with the results

reported by Just et al.

Table 4.4: InteractPlay Database Performance Results

	Protocol	#Training Subject	#Test Subject	Accuracy	ROC AUC
Just et al. IOHMM	P2	10	10	63%	-
Just et al. HMM	P2	10	10	75%	-
GT-HMM	P1	1	10	69.5%	0.745
T-HMM	P1	1	10	69.1%	0.721
LDCRF	P1	1	10	72.2%	0.785
HCRF	P1	1	10	71.8%	0.772
CRF	P1	1	10	68.9%	0.711
GT-HMM	P2	10	10	79.1%	0.831
T-HMM	P2	10	10	74.3%	0.772
LDCRF	P2	10	10	80.8%	0.857
HCRF	P2	10	10	80.1%	0.849
CRF	P2	10	10	75.3%	0.781

Results of evaluation protocol P1 report classification accuracies of 69.5%, 69.1% and 72.2% for the GT-HMM, T-HMM and LDCRF models respectively. While the performance of the models are lower than the performances reported by Just et al. the training set was comprised of gesture samples from only one signer.

Results of the evaluation protocol P2, which uses a training set of 10 subjects, show that our GT-HMM framework achieves a classification accuracy of 79.1%, a respective increase of 4.1% and 16.1% when compared to the HMMs and IOHMMs implemented by Just et al. The LDCRF and HCRF performed with classification accuracies of 80.7% and 80.1%, an improvement of 1.6% and 1% respectively when compared to the GT-HMM model. In the isolated gesture recognition experiments discussed in the previous Sections 4.5.3 - 4.5.3, the GT-HMM performs better than the LDCRF and HCRF when classifying gesture data-sets which included movement epenthesis. However,

the LDCRF and HCRF models performed better than the GT-HMM when classifying gestures from data sets which did not include movement epenthesis. Since the InteractPlay data-set does not contain movement epenthesis data, the LDCRF and HCRF results reported on the InteractPlay data-set are consistent with those in the previous sections. Thus, if the InteractPlay data-set included movement epenthesis data, we would expect the GT-HMM to robustly identify the epenthesis and therefore perform with an overall better accuracy than the LDCRF model.

Results of evaluations on the T-HMM framework, which does not use our gesture subunit initialisation, show a decrease in classification accuracy of 4.8% when compared to the GT-HMM which does use our gesture subunit initialisation. This result is a further indication of the importance of the gesture subunit initialisation technique in creating a robust HMM threshold framework which models natural human gestures.

A comparison of the results achieved on protocols P1 and P2 show that all models, including our GT-HMM model, perform with a better accuracy in user independent gesture recognition when trained on gesture samples from a larger set of subjects.

4.5.4 Continuous Gesture Recognition Experiments

Thus far we have discussed experiments conducted on isolated gestures. We now discuss experiments conducted on our GT-HMM framework and the best performing CRF model, the LDCRF model, to evaluate the performance of the respective models when spotting and classifying gestures within contin-

uous sequences of natural gestures.

In order to find candidate start and end points using the LDCRF model, we flag segments of the observations sequences which have a corresponding probability greater than a predefined threshold Ω_y. The predefined thresholds used are the best performing thresholds calculated from the ROC analysis performed on the isolated gestures discussed in Section 4.5.3. Given a continuous sequence of observations, $G = \{\mathbf{f}_1, \mathbf{f}_2, ..., \mathbf{f}_T\}$, the conditional probability $P(y|\mathbf{f}_t, \theta)$ for all observations $\mathbf{f}_t$, where $0 < t < T$, is calculated for each gesture class y. A candidate gesture, κ, is flagged for each contiguous sequence where $P(y|\mathbf{f}_t, \theta) > \Omega_y$ $(0 \leq t < T)$. In order to remove overlapping candidate gestures flagged by the LDCRF model, we use the candidate selection algorithm, described in Section 4.4.2, which utilises a gesture likelihood metric to select the final set of recognised gesture. The candidate selection metric used for the LDCRF system is defined as $\kappa_p = \frac{1}{\kappa_e - \kappa_s} \sum_{i=\kappa_s}^{\kappa_e} P(y|\mathbf{f}_i, \theta)$.

To evaluate the performance of the recognition models we test the continuous recognition models on the set of eight hand gestures, three head movements and two eyebrow gestures used in the isolated experiments in Section 4.5.3. We use the best performing models trained during the isolated recognition experiments to evaluate the continuous recognition performance of the GT-HMM framework and LDCRF model. Thus, we use an LDCRF model with 8 hidden states per label for hand gesture recognition. We use an LDCRF model with 9 hidden states per label for head gesture recognition and an LDCRF with 3 hidden states per label for eye brow gestures.

A total of 160 additional video clips of unsegmented sign language sen-

tences being performed by a fluent signer were recorded to test the performance of the continuous recognition systems. Each video clip contained at least one of the eight chosen manual signs. The three head movement gestures occurred a total of 30 times within the 160 videos while the two eye brow gestures occurred a total of 35 times. Videos were recorded at 25 frames per second with an average length of 5 seconds. In order to robustly evaluate the performance of our system, each of the 160 different sign language sentences used to test the system were performed in a mixture of different styles. The variations in the style of signs performed are similar to the variations that can occur in realistic sign language and thus testing the systems on these signs gives a good indication of how the systems will perform in real world scenarios.

Observation sequences G_L, G_R, G^H and G^E, for the left hand, right hand, head and eye brow respectively, were extracted from each video clip. Both the continuous LDCRF and GT-HMM recognition frameworks were used to process the observation sequences to spot and classify manual signs and head movement gestures. The candidate selection algorithm described in Section 4.4.2 was used to process candidate gestures for both the GT-HMM framework and LDCRF model.

In the gesture spotting and classification task, there are three types of errors: *an insertion error* occurs when the spotter reports a nonexistent gesture, *a deletion error* occurs when the spotter fails to detect a gesture, and *a substitution error* occurs when the spotter falsely classifies a gesture. From these error measures we define two performance metrics shown in Equations

4.20 and 4.21.

$$DetectionRatio = \frac{\#CorrectlyRecognizedGestures}{\#InputGestures} \tag{4.20}$$

$$Reliability = \frac{\#CorrectlyRecognizedGestures}{\#InputGestures + \#InsertionErrors} \tag{4.21}$$

Continuous Experiment Results

Tables 4.5 and 4.6 show the performance of our GT-HMM framework and the LDCRF model respectively, when spotting and classifying gestures within continuous sequences of video. The experiment shows an overall detection rate of 95.1% and an overall reliability of 87% for our GT-HMM framework and an overall detection rate of 82.7% and an overall reliability of 74.2% for the LDCRF model when spotting and classifying gestures in continuous sign language sentences.

Table 4.5: Continuous GT-HMM Performance Results

Gesture	#Correct	$\#Ins^{\ddagger}$	$\#Del^{\dagger}$	$\#Sub^{\dagger\dagger}$	Detection	Reliability	Start Error	End Error
Gone	20	2	0	0	1.0	0.9	±2.5	±8.4
Alot	20	1	0	0	1.0	0.95	±1.5	±1.6
Lost	20	2	0	0	1.0	0.9	±1.5	±3.5
Plate	19	3	1	0	0.95	0.83	±8.1	±12.2
Bike	20	1	0	0	1.0	0.95	±12.1	±12.0
Paint	20	0	0	0	1.0	1.0	±26.1	±20.7
Paper	16	1	1	3	0.8	0.76	±5.9	±1.6
Clean	18	1	1	1	0.9	0.85	±4.8	±5.2
Head Left	11	1	1	0	0.91	0.84	±10.1	±7.7
Head Right	10	2	0	0	1.0	0.83	±4.0	±4.3
Head Left Forward	8	2	0	1	0.88	0.72	±12.9	±6.5
EyeBrowDown	18	3	0	2	0.9	0.78	±19.2	±15.3
EyeBrowUp	15	2	0	0	1.0	0.88	±17.1	±24.9
Total	**215**	**21**	**4**	**7**	**0.951**	**0.87**	**±11.6**	**±9.5**

† *Number of Deletion Errors*

‡*Number of Insertion Errors*

††*Number of Substitution Errors*

Table 4.6: Continuous LDCRF Performance Results

Gesture	#Correct	$\#Ins^{\ddagger}$	$\#Del^{\dagger}$	$\#Sub^{\dagger\dagger}$	Detection	Reliability	Start Error	End Error
Gone	16	2	0	4	0.8	0.72	±8.0	±6.1
Alot	20	2	0	0	1.0	0.9	±3.0	±1.7
Lost	9	1	2	9	0.45	0.42	±3.8	±1.5
Plate	20	1	0	0	1.0	0.95	±15.7	±8.4
Bike	16	2	3	1	0.8	0.72	±12.2	±9.8
Paint	17	3	1	2	0.85	0.74	±3.0	±6.7
Paper	18	2	1	1	0.9	0.81	±0.6	±1.6
Clean	17	2	1	2	0.85	0.77	±7.1	±5.4
Head Left	11	3	0	1	0.91	0.73	±9.4	±9.2
Head Right	8	1	0	2	0.8	0.72	±18.2	±10.2
Head Left Forward	9	1	0	0	1.0	0.9	±24.3	±5.1
EyeBrowDown	14	4	0	6	0.7	0.58	±13.3	±10.2
EyeBrowUp	12	2	0	3	0.8	0.70	±14.2	±28.2
Total	**187**	**26**	**8**	**31**	**0.827**	**0.742**	**±10.1**	**±8.0**

† *Number of Deletion Errors*
‡*Number of Insertion Errors*
††*Number of Substitution Errors*

We also evaluate the performance of the start and end point detection relative to ground truth data labeled by a human sign language translator. Tables 4.5 and 4.6 show the start and end point performance results for our GT-HMM framework and the LDCRF model respectively. Each table details the average absolute difference between the spotters start and end points and the human interpreters start and end points for signs that were correctly spotted and classified. The average start and end point error for the GT-HMM framework was 11.6 frames and 9.5 frames respectively while the average start and end point error for the LDCRF model was 10.1 frames and 8.0 frames respectively.

The results of the continuous experiments show that the GT-HMM threshold model performs better than the LDCRF model. The experiments showed that our GT-HMM framework achieved an 12.4% higher detection ratio and a 12.8% higher reliability measure than the LDCRF model.

The start and end point performance experiments reveal that both the GT-HMM framework and the LDCRF model perform with similar accuracy. Since these accuracies are calculated from correctly classified gestures, the results conform with the results on the classification of isolated gestures discussed in Section 4.5.3 where the LDCRF model was shown to perform well when positively classifying gestures in the absence of epenthesis.

Continuous User Independent Experiment Results

In this section we conduct a second continuous sign language recognition experiment. We evaluate the same set of continuous recognition models,

used in the user dependent experiments, on a set of sign language sentences performed by a second signer not used in the training set. Figure 4.21 shows an example of the eight signs being performed by the second signer.

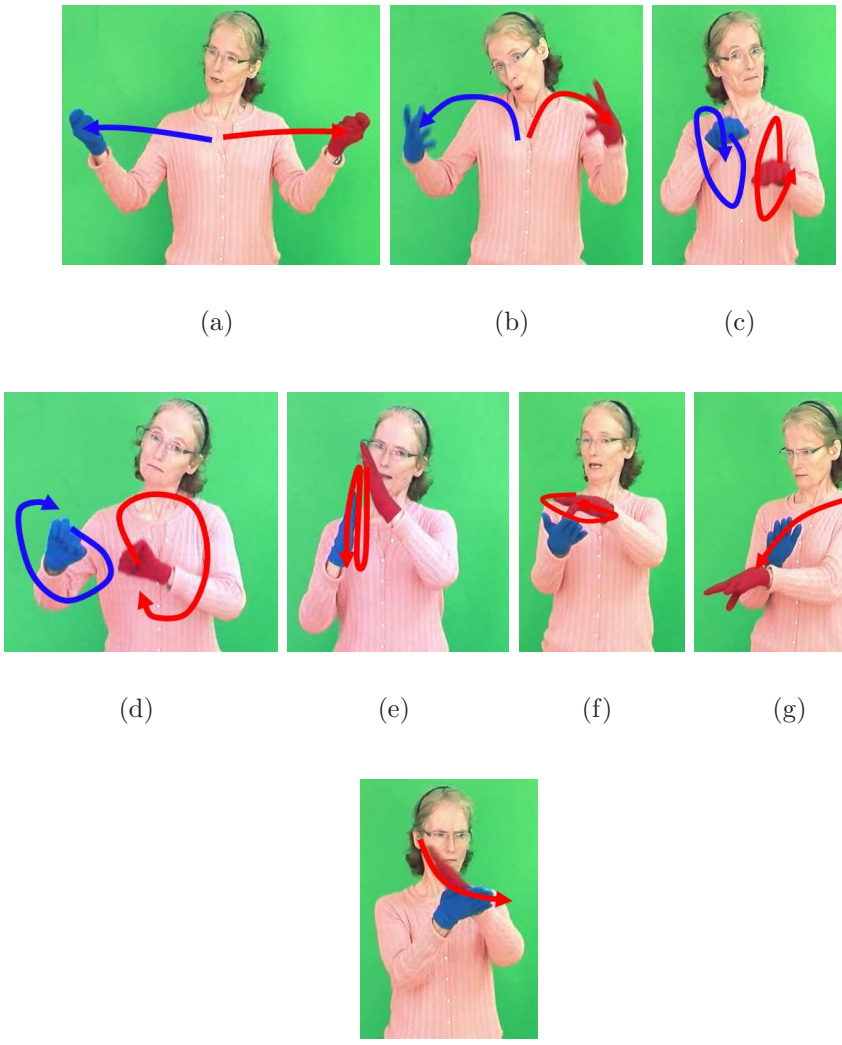
A total of 160 additional video clips of unsegmented sign language sentences being performed by the second signer were recorded. Each video clip contained at least one of the eight chosen manual signs.

Observation sequences, G_L and G_R, for the left hand and right hand respectively, are extracted from each video clip. Both the continuous GT-HMM and LDCRF recognition frameworks are then used to process the observation sequences to spot and classify signs from the videos of the second signer.

Tables 4.7 and 4.8 show the performance of our GT-HMM framework and the LDCRF model respectively when recognising signs performed by the second signer. The experiment shows an overall detection rate of 80.4% and an overall reliability of 70.2% for our GT-HMM framework and an overall detection rate of 67.6% and an overall reliability of 57.8% for the LDCRF model.

When compared to the detection rates achieved in the user dependent experiments, the overall detection rate of the GT-HMM framework drops by 15.2% and, similarly, the overall detection rate of the LDCRF drops by 15.5%. These results show that our GT-HMM framework consistently performs with a higher gesture detection rate than that of the LDCRF when recognising signer-independent signs.

In previous works which have used a small number of signers in the training set, results of user independent recognition evaluations have seen large



(a) (b) (c)

(d) (e) (f) (g)

(h)

Figure 4.21: Example of the eight different performed by Signer 2 (a) Newspaper, (b) A lot, (c) Bike, (d) Clean, (e) Paint, (f) Plate, (g) Lost, (h) Gone

decreases when compared to user dependent recognition results [OR05]. For example, in Assan et al. [AG98], accuracy for training on one signer and testing on another was 51.9% compared to 92% when the same signer supplied both training and test data. As shown in the experiments on the Interact-Play data-set, user independent recognition performance using our GT-HMM threshold framework can be improved by utilising a larger number of subjects. While the experiments discussed in this section show a decrease in the signer-independent detection rate, the fact that only one signer was represented in the training set suggests that a decrease of only 15.2% can be interpreted as a promising result.

Table 4.7: User Independent Continuous GT-HMM Performance Results

Gesture	#Correct	$\#Ins^{\ddagger}$	$\#Del^{\dagger}$	$\#Sub^{\dagger\dagger}$	Detection	Reliability	Start Error	End Error
Gone	10	3	0	7	0.58	0.5	6.3	2.8
Alot	18	4	0	8	0.69	0.6	2.2	1.2
Lost	16	3	0	2	0.88	0.61	1.5	3.5
Plate	12	4	2	2	0.75	0.6	9.6	13.1
Bike	28	2	1	2	0.90	0.84	3.0	8.3
Paint	20	2	0	0	1.0	0.9	6.1	4.0
Paper	15	3	1	5	0.71	0.62	5.1	3.6
Clean	13	3	0	2	0.86	0.72	7.9	8.3
User Independent Total	**132**	**24**	**4**	**28**	**0.804**	**0.702**	**5.2**	**5.6**
User Dependent Total	**153**	**11**	**3**	**4**	**0.956**	**0.894**	**6.6**	**8.1**

† *Number of Deletion Errors*
‡*Number of Insertion Errors*
††*Number of Substitution Errors*

Table 4.8: User Independent Continuous LDCRF Performance Results

Gesture	#Correct	$\#Ins^{\ddagger}$	$\#Del^{\dagger}$	$\#Sub^{\dagger\dagger}$	Detection	Reliability	Start Error	End Error
Gone	9	4	0	8	0.52	0.36	4.2	2.5
Alot	22	2	0	4	0.84	0.78	5.0	2.3
Lost	9	2	0	9	0.5	0.45	3.4	5.5
Plate	11	5	1	4	0.68	0.52	8.9	14.1
Bike	27	5	0	4	0.87	0.75	10.8	2.6
Paint	13	3	0	7	0.65	0.56	6.6	4.5
Paper	8	3	2	11	0.38	0.33	5.9	1.6
Clean	12	4	1	2	0.8	0.63	4.8	5.2
User Independent Total	**111**	**28**	**4**	**49**	**0.676**	**0.578**	**6.2**	**10.9**
User Dependent Total	**133**	**15**	**8**	**19**	**0.831**	**0.76**	**6.6**	**8.1**

† *Number of Deletion Errors*
‡*Number of Insertion Errors*
††*Number of Substitution Errors*

4.5.5 Multimodal Recognition Examples

Thus far we have discussed how our techniques can be used to spot different modalities of sign language communication. Incorporating non-manual signals such as eyebrow gestures and head movement gestures into a sign language recognition framework provides the necessary foundations for differentiating between different types of questions, and also recognising the start of sign language sentences. In this section we show gesture spotting results from a number of manual and non-manual signals and discuss how the combination of these signals can be utilised to create a more comprehensive understanding of sign language sentences.

Figure 4.22 shows the gestures spotted by our system in three different sentences. Figure 4.22(a) and Figure 4.22(b) show gestures spotted from two sentences where the signer performs the signs "CAR PETROL ALL GONE" in both sentences.

In the first sentence the signer is asking a question "CAR PETROL ALL GONE HOW?", but in the second sentence the signer is asking a yes/no question "CAR PETROL ALL GONE?". The manual signs for both these sentences are the same but the difference can only be recognised from the head movement and eyebrow gestures. It can be seen in the first sentence that our system spots an eyebrow down gesture coinciding with a left head movement followed by right head movement. This suggests that the signer is asking a "wh" question. In the second sentence our system spots an eyebrow down gesture at the beginning of the sentence followed by a head forward movement, indicating the signer is asking a yes/no question. Figure 4.22(c)

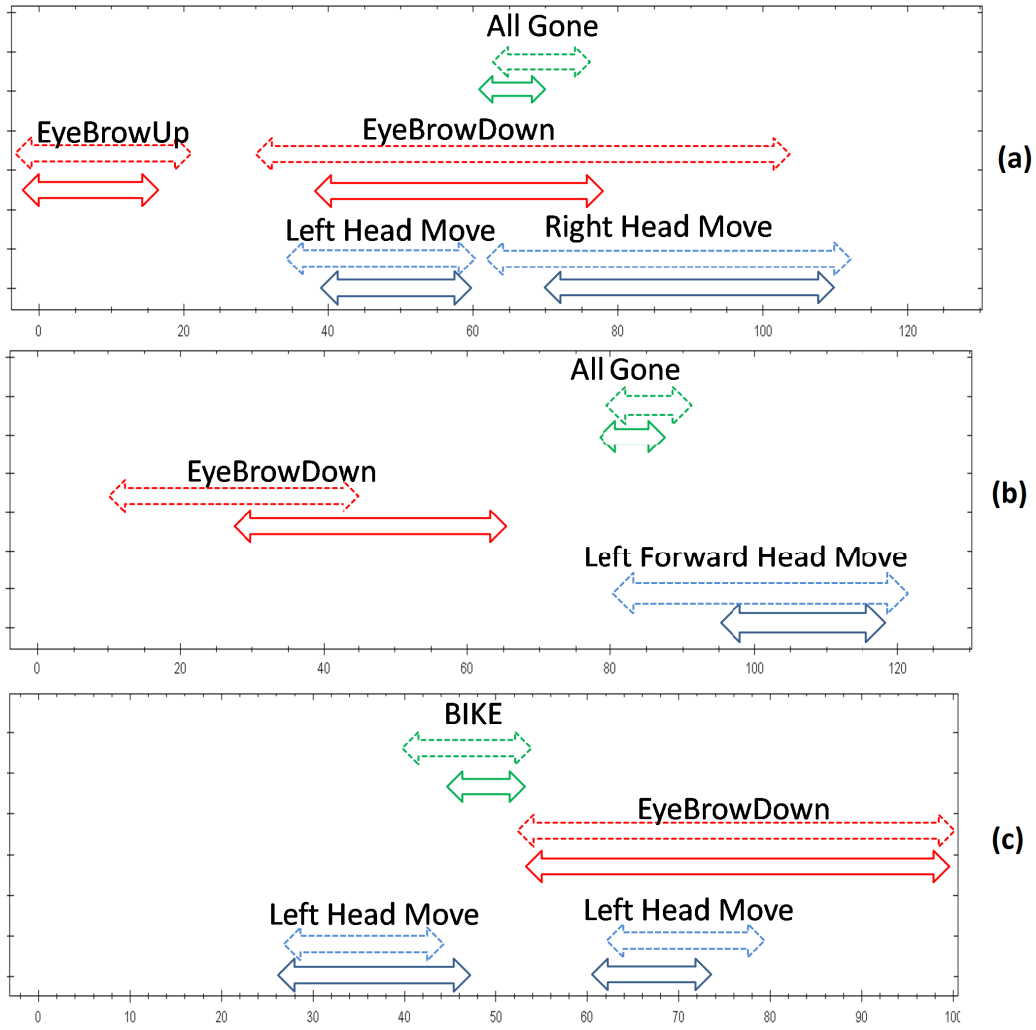


Figure 4.22: Multimodal gesture labeling comparison of a human interpreter vs. our recognition system (Dotted Arrows Represent Hand Labeled Gestures while solid arrows represent labels generated by our system)

shows gestures from a sentence "WHO BIKE BROKE?", where our system spots an eyebrow down gesture coinciding with a left head movement. Similar to the gestures in Figure 4.22(a), the eyebrow down gesture coinciding with a head movement gesture indicates a "wh" question. Also from Figure 4.22 (a), it was observed that an eyebrow up gesture occurred at the start of the sequence. This is an interesting observation as the eyebrow up gesture is linked to the start of a new sentence or sequence.

4.6 Conclusion

In Section 2.2.2, we discuss current methods of continuous gesture recognition. The disadvantage of the majority of these methods is that explicit training of movement epenthesis models are required or unnatural constraints are put on the signer, such as unnatural pauses between words. The main contribution of the work proposed in this Chapter is that we develop a robust framework for the recognition of spatiotemporal based gestures which does not require explicit epenthesis training or specific rules to determine sign boundaries. Our framework requires only that the dedicated gesture models be trained and as a result of this training a single epenthesis model can be implemented. We expand on the work of Lee and Kim [LK99] to develop a GT-HMM framework, utilising our novel gesture subunit initialisation technique, which models continuous multidimensional gesture observations within a parallel HMM network.

We evaluate our GT-HMM framework and compare it to current models for recognising human motion. HMMs, CRFs, HCRFs and LDCRFs have

recently been implemented in systems for automatically recognising different human actions. We evaluate these techniques in the domain of sign language gesture recognition. In order to evaluate the performance of the models when recognising sign language gestures, it is important to evaluate each model when identifying movement epenthesis as well as evaluating the performance of the models when classifying gestures. The first set of experiments were conducted on an isolated data set of motion based manual signs and on an isolated data set of non-manual head motion and eye brow motion gestures.

In experiments carried out on all three gesture types, the best performing model was the LDCRF when tested on a data set which did not include movement epenthesis. The results of this experiment are consistent with previous experiments on HCRFs and LDCRFs in which Wang et al. [WQM+06] and Morency et al. [MQD07] showed that HCRFs and LDCRF performed better than the standard HMM model when classifying gestures. When a data set which included movement epenthesis was introduced to the experiments the performance of the standard HMM model, and all CRF models, dropped significantly in relation to the performance of GT-HMM model. The GT-HMM model performed best in all three experiments with movement epenthesis data. The GT-HMM achieves an AUC of 0.976, 0.936 and 0.948 for the hand gesture, head gesture and eye brow gesture evaluations respectively. In comparison, the LDCRF model achieves an AUC of 0.942, 0.894 and 0.918 for the hand gesture, head gesture and eye brow gesture evaluations respectively. These results suggest that there is an inherent weakness in the LDCRF model when identifying movement epenthesis. As the number and

variability in style of epenthesis increases, we would expect the performance of the LDCRF model to further decrease. To test this we further evaluate the GT-HMM and LDCRF models when spotting and classifying gestures from continuous sign language sentences. Results of the continuous experiments show that our GT-HMM framework achieves a 12.4% higher detection ratio and a 12.8% higher reliability measure than the LDCRF model when tested on 220 different hand, head and eye brow gestures.

Additionally, a set of signer-independent experiments are conducted to evaluate the performance of the models when recognising signs performed by a signer not represented in the training set. Signer-independent evaluations conducted on the InteractPlay data-set, which contains no epenthesis data, show that the GT-HMM and LDCRF models performs best when compared to HMMs and IOHMMs implemented by Just et al. [JM09]. Furthermore, signer-independent evaluations on continuous sign language sentences show that our GT-HMM framework consistently performs with a higher gesture detection rate than that of the LDCRF. The overall detection rate of the GT-HMM and LDCRF models decrease by only 15.2% and 15.5% respectively, when compared to signer-dependent results.

An important aspect of our GT-HMM framework is the gesture subunit initialisation technique which is used to automatically initialise the parameters of the GT-HMM framework. We conduct experiments on manual and non-manual data sets and show that initialising the HMMs using our technique consistently improves hand gesture, head gesture and eye brow gesture classification performance by 3.5%, 6.3% and 4.3% respectively when com-

pared to a standard initialisation technique. Furthermore, experiments on the Interactplay data-set also indicate that our gesture subunit initialisation improves classification performance, showing a 4% increase in performance.

We also discuss the importance of non-manual signals in sign language. In Section 2.2.2, we highlight that there are a limited number of works which incorporate both manual and non-manual signals into a single framework for continuous automatic sign language recognition. Another contribution of this work is that we have presented an approach towards multimodal interpretation of ISL sentences. Unlike previous gesture spotting techniques which utilise explicit segmentation models, the spatiotemporal gesture recognition framework we propose is not specific to any particular type of gesture and we demonstrate this by showing that manual and non-manual signals can be robustly spotted and classified from within continuous sign sequences. By incorporating non-manual signals such as eyebrow gestures and head movement gestures into our framework, we show that our technique provides the necessary foundations for differentiating between different types of questions and also recognising the start of sign language sentences. Also unlike current works, each manual and non-manual signal is processed independently within our multimodal framework.

One limitation of the continuous recognition techniques we propose in the chapter is the scalability of the method to handle very large vocabulary sets. In the experiments we carry out in this chapter, we evaluate the system on, at most, eight gesture classes. As the number of gesture classes grows significantly, the speed at which continuous gesture spotting can be carried

out will slow down significantly. A solution to this may be to reconsider the sliding window approach to the continuous gesture spotting in an attempt to reduce the number of times the Viterbi algorithm is run per frame.

The techniques we present in this Chapter have a significant impact on the overall goal of full sign recognition. Our techniques enable the robust learning of signs and movement epenthesis with minimal sign labeling. The learned models are then capable of accurately spotting signs and classifying movement epenthesis from within continuous sign language video streams. An advantage of being able to robustly learn models for signs and movement epenthesis with minimal hand labeling is that the models can be implemented in a weakly supervised learning framework in order to learn signs automatically. The ability of our GT-HMM framework to be implemented in a weakly supervised sign language learning system will be discussed in Chapter 5.

Chapter 5

Weakly Supervised Training [1]

5.1 Introduction

Sign language recognition is the task of detecting and classifying signs in a signed sentence, in a set vocabulary. The overall goal of the work detailed in this thesis is to develop a fully automated framework for the training and recognition of sign language. The main contribution of the work detailed in this chapter is that we propose a weakly supervised system, using our novel Multiple Instance Learning (MIL) density matrix algorithm, which automatically extracts isolated samples of signs which can then be used to automatically train our hand gesture models. We propose techniques to incorporate our hand posture recognition system, proposed in Chapter 3, and our spatiotemporal recognition framework, proposed in Chapter 4, into

[1]The completed work discussed in this chapter has been published in the following journal: **D. Kelly**, J. Mc Donald and C. Markham,"Weakly Supervised Training of a Sign Language Recognition System using Multiple Instance Learning Density Matrices", In IEEE Transactions on Systems, Man and Cybernetics-Part B

a single recognition framework which automatically learns and recognises signs. Currently there exists no work which deals with all of these problems in a sign language recognition system.

5.1.1 System Overview

It is the goal of this research to develop a weakly supervised system which automatically extracts isolated samples of signs which can then be used to train our temporal gesture and hand posture recognition frameworks. Section 5.4 presents our automatic sign extraction system which utilises the text translations of the videos to automatically extract isolated samples of target words. Section 5.3 presents the distance functions we developed as part of our automatic sign extraction. Section 5.5 presents our framework for training classifiers which model the automatically extracted signs. Section 5.6 presents an extended version of our continuous sign recognition system, detailed in Section 4.5.4, which utilises the automatically trained temporal and hand posture classifiers. Section 5.7 presents experimental results. Figure 5.1 shows a visualisation of the flow of our proposed automatic training and sign recognition systems.

5.2 Feature Extraction

In this chapter we will evaluate our MIL density matrix algorithm and full sign recognition system using an extended version of the ISL data set used in Chapter 4. As discussed in the ISL experiments section of Chapter 4, tracking

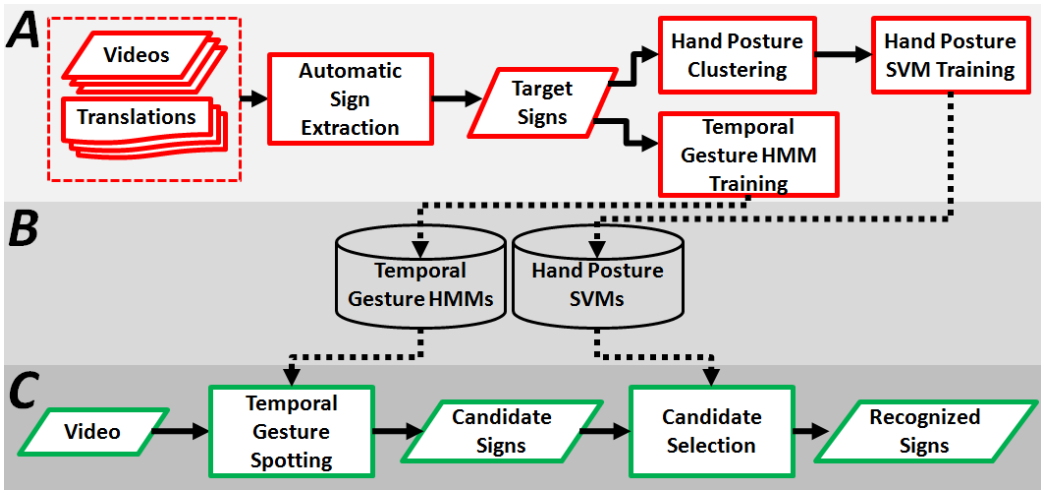


Figure 5.1: Flowchart of our proposed automatic sign training and recognition framework (A) System Training, (B) Sign Classifiers, (C) Sign Recognition

of the hands was performed by tracking coloured gloves using the Mean Shift algorithm [CRM00] (see Figure 5.2). As discussed in Chapter 3, the key feature used for the analysis of the hand shape is the external contour, C, made by the hand (see Figure 5.2(b)). The dominant hand is used to convey most hand shape information and thus we consider only the shape of the dominant hand when analysing hand shape. When the dominant and the non-dominant hands overlap, we extract a single contour made around the dominant and non-dominant hand as illustrated in Figures 5.2(a) and 5.2(b).

5.3 Gesture Similarity Functions

Before describing our MIL density matrix algorithm which we utilise to automatically extract isolated samples of sign language target words, we first

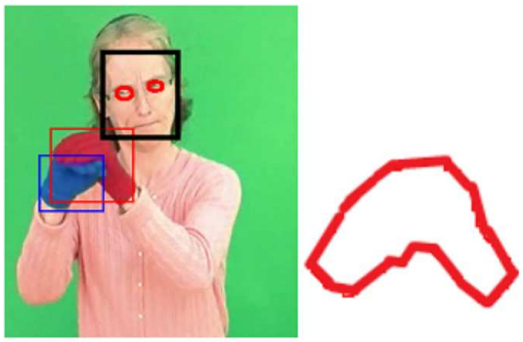


Figure 5.2: Extracted Features from Image (a) Feature Positions (b) Hand Contour

describe the gesture similarity functions used as part of the MIL algorithm.

In order to compare temporal gestures and hand postures we require a measure of similarity between gesture frames. We implement two similarity functions, one for temporal gesture distances and one for hand posture distances.

5.3.1 Temporal Gesture similarity Function

As discussed in Chapter 4, the best performing feature vector for classifying hand gestures was the feature vector $\widehat{\mathbf{f}} = \{RP_x, RP_y, V_x, V_y, D_H\}$, where RP_x and RP_y describes the position of the hands relative to the eyes, V_x and V_y describes the direction of the movement of the hand and D_H describes the distance between the two hands. This is calculated for both right and left hands, which we denote as $\widehat{\mathbf{f}}_R$ and $\widehat{\mathbf{f}}_L$ respectively.

Given two feature vectors $\widehat{F}$ and $\widehat{F}'$, where $\widehat{F} = \{\widehat{\mathbf{f}}_R, \widehat{\mathbf{f}}_L\}$, we require a measure of similarity between the two features. In order to calculate this

similarity, we define the similarity function $D^G()$ as the weighted sum of the distance for the right and the left hand where ω_R and ω_L define the right and left weights respectively:

$$D^G(\widehat{F}, \widehat{F}') = \omega_R\Big(1 - d^G(\widehat{f}_R, \widehat{f}'_R)\Big) + \omega_L\Big(1 - d^G(\widehat{f}_L, \widehat{f}'_L)\Big) \tag{5.1}$$

The distance between individual hands $d^G()$ is then calculated by first calculating three individual distances between each of the three spatiotemporal gesture features (movement, position, hand distance) as follows:

$$d^G_{pos}(\widehat{f}, \widehat{f}') = \sqrt{(RP_x - RP'_x)^2 + (RP_y - RP'_y)^2} \tag{5.2}$$

$$d^G_{mov}(\widehat{f}, \widehat{f}') = \sqrt{(V_x - V'_x)^2 + (V_y - V'_y)^2} \tag{5.3}$$

$$d^G_{dis}(\widehat{f}, \widehat{f}') = \sqrt{(D_h - D'_h)^2} \tag{5.4}$$

Using each of the three distances, the overall distance can then be calculated using a normalized sum of the three distances:

$$d^G(\widehat{f}, \widehat{f}') = \omega_{pos} d^G_{pos}(\widehat{f}, \widehat{f}') + \omega_{mov} d^G_{mov}(\widehat{f}, \widehat{f}') + \omega_{dis} d^G_{dis}(\widehat{f}, \widehat{f}') \tag{5.5}$$

Where $\omega_x = \frac{1}{Max^G_x * 2} (x \in \{pos, mov, dis\})$ and Max^G_x is the max distance $d^x()$ calculated between all pairs of hand postures in the data set. The scale factors ω_x normalise each distance measure such that $0 \leq d^G(\widehat{f}, \widehat{f}') \leq 1$ for

all pairs of features $\widehat{f}$ and $\widehat{f}'$.

5.3.2 Hand Posture Similarity Function

The overall similarity between two hand contours, C and C', is calculated as defined in Equation 5.6.

$$D^H(C, C') = 1 - \left(\omega_{Sf} d^{SF}(C, C') + \omega_{Hu} d^{hu}(C, C')\right) \tag{5.6}$$

Where $d^{SF}(C,C')$ and $d^{Hu}(C,C')$ are the eigenspace Size Function distance and the Hu Moment distance functions, defined in Chapter 3, respectively. We define $\omega_x = \frac{1}{Max_x^H * 3} (x \in \{Sf, Hu\})$ and Max_x^H as the max distance $d_x^H()$ calculated between all pairs of frames in the data set. The scale factors ω_x normalise each distance measure such that $0 \leq D^H(C,C') \leq 1$ for all pairs of hand contours C and C'.

5.4 Automatic Sign Extraction

In order to train the hand posture and spatiotemporal recognition models proposed in Chapters 3 and 4, multiple training samples of target concepts must be labeled and extracted from unconstrained sign language sentences. The labeling of sign data is an integral step in modeling each sign concept, thus, in order to create valid training samples, the sign data must be labeled consistently by a fluent signer. The labeling of spatiotemporal segments of signs consists of marking the start and end point of target signs within a continuous sentence. Following this, hand posture labeling must be carried out

by marking key hand postures within the spatiotemporal segments. Manually performing this labeling process is an un-intuitive and time consuming process for fluent signers to label in a consistent manner. The most consistent, and freely available, type of sign language labeling is sentence text translations where each sign language sentence is translated to written English. The problem with text translations labels is that the labels are at a sentence level and not at the required frame level. The aim of the work discussed in this chapter is to develop techniques to automatically extract frame level labels using sign language text translations.

Given a target word, a set of video sequences and corresponding weakly aligned text translations, the goal is to automatically identify which frames, if any, contain the target word. Furthermore, we must also identify which frames contain key hand postures related to the target word. The difficulty in this problem is that no one-to-one mapping exists between English translations and the signs since the ordering of sign language is different to English translations. Another difficulty with the learning task is that there exists ambiguities in the translation task where the same sign may have different translations or a word may appear in the text translation but the corresponding sign may not occur in the video. This introduces a significant correspondence problem between the translation and overlapping video sequence which means that the supervision is weak.

In order to find frame level labels we can formulate the task as a MIL problem. MIL is a weakly supervised variation on supervised learning for problems with incomplete knowledge about labels of training examples. In-

stead of receiving a set of instances which are labeled positive or negative, the learner receives a set of bags that are labeled positive or negative. Each bag contains many instances. A bag is labeled negative if all the instances in it are negative. On the other hand, a bag is labeled positive if there is at least one instance in it which is positive. From a collection of labeled bags, the learner tries to induce a concept that will label individual instances correctly.

Using MIL notation, we define a video sequence as a bag, where the set of positive bags, $B^+ = \{B_0, ..., B_{N_B}\}$, is the set videos which the target word occurs and N_B is the number of bags in the set of bags B^+. The videos contained within the set of bags should be chosen such that the target word appears with most frequency compared to other words in the text translations. This can be done automatically by a process of word counting.

Each bag B_i represents a sequence of features for each frame $B_i = \{B_i[0], ..., B_i[N_{Bi}]\}$. To reduce the impact of translation ambiguities we develop a Multiple Instance Learning solution that requires only positive bags. In a traditional MIL framework, a learner uses both positive and negative bags to learn the intended concept, where a negative bag represents a sequence of features which do not lie within the concept. In the case of sign language videos, a negative bag would represent a set of videos which did not contain the text translation for a target word. Since there are ambiguities in the translations, there is no guarantee that the corresponding sign occurs in the videos. The work of Buehler et al. [BZE09] acknowledges the difficulty in estimating errors in negative bags without manual labeling and their work uses a heuristic to determine errors in the negative bags. We address the problem of errors in

negative bags by developing a MIL framework which requires positive bags only. This eliminates the need to automatically identify errors in negative bags and thus limits the type of translation ambiguities, which our system must deal with, to errors in positive bags. Translation ambiguities which our system must identify is thus limited to situations were the text translation contains a target word but the corresponding sign does not occur in the video.

5.4.1 MIL Density Matrix

We develop a novel MIL density matrix algorithm, inspired by diverse density MIL [MLPpE98], to label videos at frame level. Since hand postures and temporal gestures are independent channels, we propose a multi-channel MIL density matrix algorithm. We define sperate bags for each channel and denote the set of temporal gesture bags as G^+ and denote the set of hand shape bags as H^+.

The first step in our MIL Density Matrix algorithm is to compute comparison matrices $\overline{G_{ij}}$ and $\overline{H_{ij}}$ in order to compare each pair of bags (G_i,G_j) and (H_i,H_j) respectively. Each comparison matrix $\overline{G_{ij}}$ and $\overline{H_{ij}}$ corresponds to a $N_{Bj}\times N_{Bi}$ matrix.

For each column t ($0 \leq t < N_{Bi}$) in the temporal gesture comparison matrix $\overline{G_{ij}}$, we store only the most similar comparison between $G_i[t]$ and all features in G_j. All other entries in that column are set to zero. Each element of the comparison matrix, $\overline{G_{ij}}[t, \iota^t]$, is calculated as defined in Equations 5.7 and 5.8, where ι^t is the frame index of G_j which is most similar to $G_i[t]$.

Figure 5.3 shows a visual example of the calculation of the spatiotemporal comparison matrix where we show the calculation of the similarity measures for column 0 by calculating all similarities $\left[D^G(G_i[0],G_j[0]),...,D^G(G_i[0],G_j[8])\right]^T$. In column 0, it can be seen that the index, ι^0, representing the most similar gesture frame is the 4^{th} frame of G_j. The 4^{th} element of the first column in the comparison matrix $\overline{G_{ij}}$ then stores the similarity measure between $G_i[0]$ and $G_j[4]$ while all other entries in that column are set to zero.

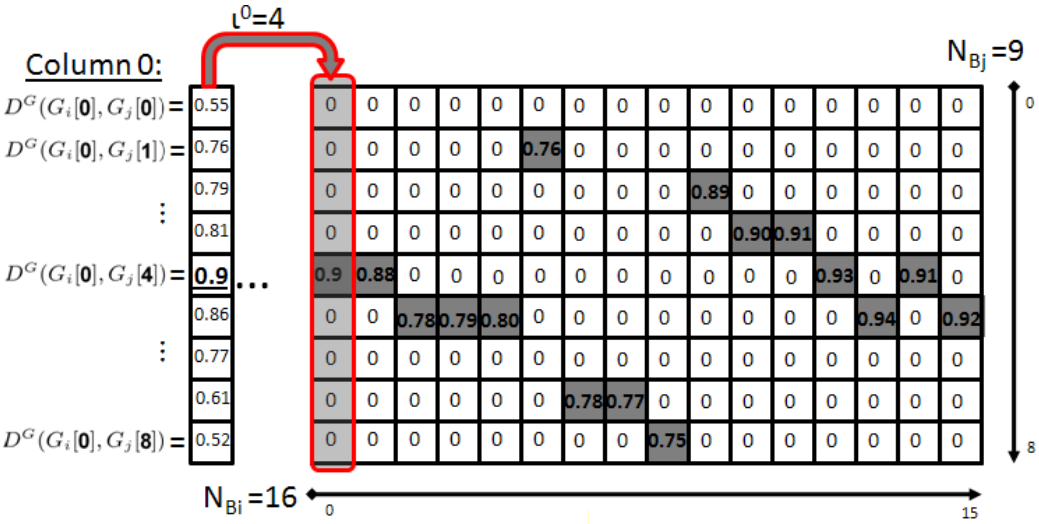


Figure 5.3: Visual representation of the calculation of the spatiotemporal gesture comparison matrix $\overline{G_{ij}}$

The hand posture comparison matrix, $\overline{H_{ij}}$, is calculated in a similar fashion. For each column t ($0 \leq t < N_{Bi}$) in the hand posture comparison matrix, $\overline{H_{ij}}$, we store the similarity between $H_i[t]$ and $H_j[\iota^t]$, where ι^t is the frame index of G_j which is most similar to $G_i[t]$ (as defined in Equation 5.9). For example, in Figure 5.3, it was identified that the most similar gesture frame for column 0 was the 4^{th} frame of G_j. This means that in the corresponding

hand posture comparison matrix, $\overline{H_{ij}}$, the 4^{th} element of the first column stores the similarity measure between $H_i[0]$ and $H_j[4]$ while all other entries in that column are set to zero. Using the most similar spatiotemporal comparison index allows us to prioritise spatiotemporal gestures. This allows us to match hand poses which help us to discriminate between spatiotemporal gestures that have similar spatiotemporal patterns but differ only in hand pose.

Each element of the comparison matrix is calculated using the similarity functions $D^G()$ and $D^H()$ which we define in Section 5.3. It should be noted that the similarity functions $D^G()$ and $D^H()$ are implemented such that similar frames return values close to 1 while frames which are not similar return values close to 0. The threshold values τ^G and τ^H are implemented such that the similarity comparisons $\overline{G_{ij}}[t, \iota^t]$ and $\overline{H_{ij}}[t, \iota^t]$ only include video frames which are within a set similarity. This reduces the effect non similar frames have on the overall detection of target words and, through experiments, we have found that a threshold value of 0.75, for both τ^G and τ^H, has performed well.

$$\iota^t = \underset{0 \leq \tilde{\iota} < N_{Bj}}{\operatorname{argmax}} \quad D^G(G_i[t], G_j[\tilde{\iota}]) \tag{5.7}$$

$$\overline{G_{ij}}[t, \iota^t] = \begin{cases} D^G(G_i[t], G_j[\iota^t]) & \text{if } D^G(G_i[t], G_j[\iota^t]) > \tau^G; \\ 0 & \text{else.} \end{cases} \tag{5.8}$$

$$\overline{H_{ij}}[t, \iota^t] = \begin{cases} D^H(H_i[t], H_j[\iota^t]) & \text{if } D^H(H_i[t], H_j[\iota^t]) > \tau^H; \\ 0 & \text{else.} \end{cases} \tag{5.9}$$

For convenience we denote a bag as B below, but the following computations are calculated for both temporal gesture bags G and hand pose bags H by replacing notation B and $\mathbf{b}$ with H and $\mathbf{h}$ or G and $\mathbf{g}$ accordingly.

After computing the comparison matrix $\overline{B_{ij}}$, we convert it to a comparison vector $\overline{\mathbf{b}_{ij}}$ as shown in Equation 5.10. Where each vector $\overline{\mathbf{b}_{ij}}$ corresponds to a N_{Bi}-dimensional row vector. Each comparison vector is used as a measure of how closely matched each frame, in bag B_i, is to the most similar frame in bag B_j.

$$\overline{\mathbf{b}_{ij}}[t] = \sum_{n=0}^{N_{Bj}} \overline{B_{ij}}[t, n] \tag{5.10}$$

Each element $\overline{\mathbf{b}_{ij}}[t]$ represents the sum of the similarity metrics in the t^{th} column of the comparison matrix $\overline{B_{ij}}$. Since the t'th column in $\overline{B_{ij}}$ contains, at most, one similarity metric, then the element of comparison vector $\overline{\mathbf{b}_{ij}}[t]$ corresponds to the same similarity metric.

We also define a transposed comparison vector, as shown in Equation 5.11, where each element, $\overline{\mathbf{b}_{ij}}'[t]$, of the N_{Bj}-dimensional column vector represents the sum of the similarity metrics in the t^{th} row of the comparison matrix. Each transposed comparison vector is used as a measure of which frames, in B_j, has the most in common with the bag B_i.

$$\overline{\mathbf{b}_{ij}}'[t] = \sum_{n=0}^{N_{Bi}} \overline{B_{ij}}[n, t] \tag{5.11}$$

Each comparison vector $\overline{\mathbf{b}_{ij}}$ represents a comparison between the i^{th} and j^{th} bags. The comparison vector does not have any representation of how

the i^{th} and j^{th} bags interact with all other bags in the set of bags. In order to build a full comparison model between each pair of bags (B_i,B_j), we must not only understand how B_i and B_j interact with each other, but also how the pair of bags (B_i,B_j) interact with all other bags in the set of bags B^+. In order to do this, we first build a model of B_i compared to all other bags followed by a model of B_j compared to all other bags. We then use both these models to build a model of how B_i and B_j interact. We compare B_i to all other bags by summing all comparison vectors $\overline{\mathbf{b}_{ik}}, \forall k \in B^+$ (see Equation 5.12). We then compare B_j to all other models by summing all transposed comparison vectors $\overline{\mathbf{b}_{kj}}^{'}, \forall k \in B^+$ (see Equation 5.13).

$$\mathbf{b}_i = \sum_{k=0}^{N_B} \overline{\mathbf{b}_{ik}} \tag{5.12}$$

$$\mathbf{b}_j^{'} = \sum_{k=0}^{N_B} \overline{\mathbf{b}_{kj}}^{'} \tag{5.13}$$

Where $\mathbf{b}_i$ corresponds to a N_{Bi}-dimensional row vector and $\mathbf{b}_j^{'}$ corresponds to a N_{Bj}-dimensional column vector. Figure 5.4 shows the temporal gesture comparison matrices, $\overline{G_{ij}}$, for a sample set of 5 positive bags for the target sign "Alot". Each matrix, at co-ordinates (G_i,G_j), corresponds to a visual representation of comparison matrices $\overline{G_{ij}}$, where dark pixels represent a high similarity between gesture frames. The hand labeled time segments in each bag represent frames of each video which were hand labeled as being a target sign. Figure 5.4 also shows a visualisation of summed comparison vectors, $\mathbf{g}_1$ and $\mathbf{g}_5^{'}$, being computed from the corresponding set of comparison

matrices.

It can be seen from the density matrices that there exists a high density of similar frames where the hand labeled sections occur, illustrating the power of our technique for measuring areas of similarity.

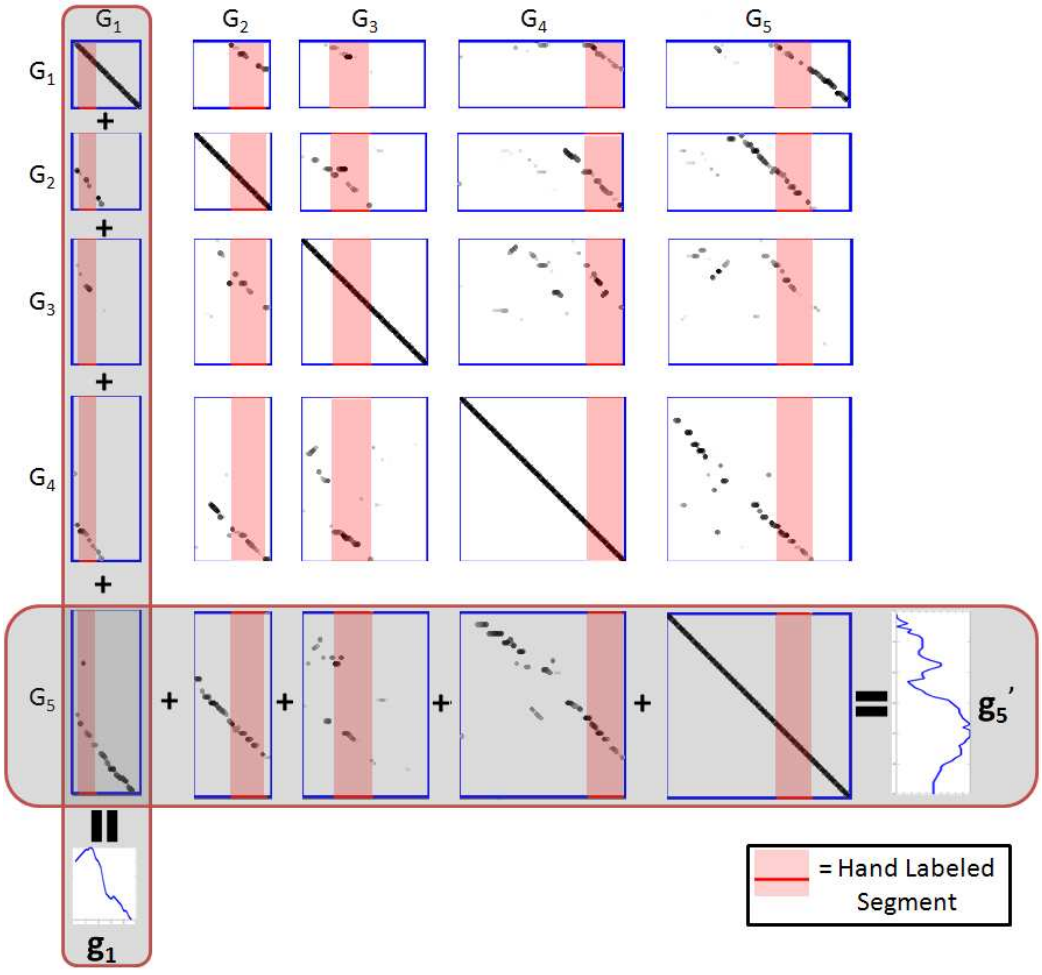


Figure 5.4: Visualisation of comparison matrices $\overline{G_{ij}}$ and calculation of summed comparison vectors $\mathbf{g}_1$ and $\mathbf{g}_5^{'}$

Following the calculation of the summed comparison vectors, we then

compute the density matrix $\Phi(B_i, B_j)$, which measures the interaction between the pair of bags B_i and B_j, by multiplying the summed comparison vectors to calculate a $N_{Bj} \times N_{Bi}$ matrix:

$$\Phi(B_i, B_j) = \mathbf{b}_i \mathbf{b}_j^{'} \tag{5.14}$$

Where $\mathbf{b}_i$ corresponds to a N_{Bi} row vector and $\mathbf{b}_i$ corresponds to a N_{Bj} column vector. Figure 5.5 shows a visualisation of the computation of the density matrix $\Phi(G_1, G_5)$ computed from the summed comparison vectors shown in Figure 5.4.

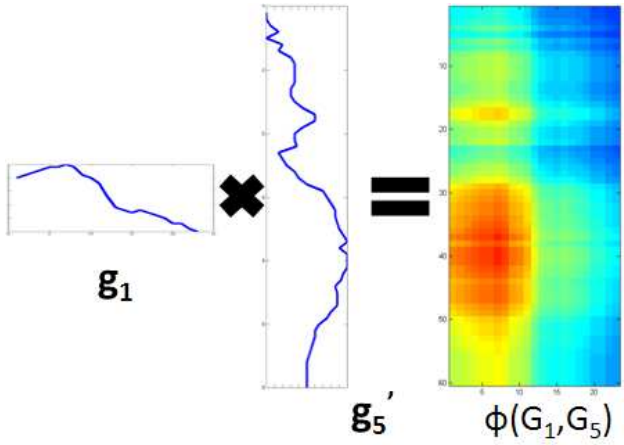


Figure 5.5: Visualisation of density matrix $\Phi(G_1, G_5)$

We then compute a density vector $\Psi(B_i)$ for bag B_i using the set of density matrices $\{\Phi(B_i, B_1), ..., \Phi(B_i, B_{N_B})\}$. Each element, $\Psi(B_i)[t]$, of the density vector is calculated by averaging the t^{th} column of each of the density matrices $\Phi(B_i, B_j)$:

$$\Psi(B_i)[t] = \frac{1}{N_B} \sum_{j=0}^{N_B} \sum_{k=0}^{N_{Bj}} \frac{\Phi(B_i, B_j)[t,k]}{N_{Bj}} \tag{5.15}$$

An overall density vector, Ψ_i, is then calculated from the inner product of the hand posture and temporal gesture density vectors:

$$\Psi_i \quad = \quad \Psi(H_i) \cdot \Psi(G_i) \tag{5.16}$$

Where $\Psi(H_i)$ and $\Psi(G_i)$ denotes the density vector, of the i'th bag, for hand posture and temporal gestures respectively.

Figure 5.6 shows a further example of the temporal gesture comparison matrices, $\overline{G_{ij}}$, computed from 12 positive bags of the target sign "Play" (2 of which contain translation ambiguities). Each matrix, at co-ordinates (G_i,G_j), corresponds to a visual representation of comparison matrices $\overline{G_{ij}}$, where dark pixels represent a high similarity between gesture frames.

5.4.2 Automatic Sign Labeling

Given the density vector Ψ_i we now label frames in the corresponding video.

In order to account for translation ambiguities we must detect whether or not the target sign was actually performed in the video even though the translation information specifies that it does occur. In order to do this, we flag video sequences which have no frames with high similarity when compared to other video sequences in the same positive bag. We first apply

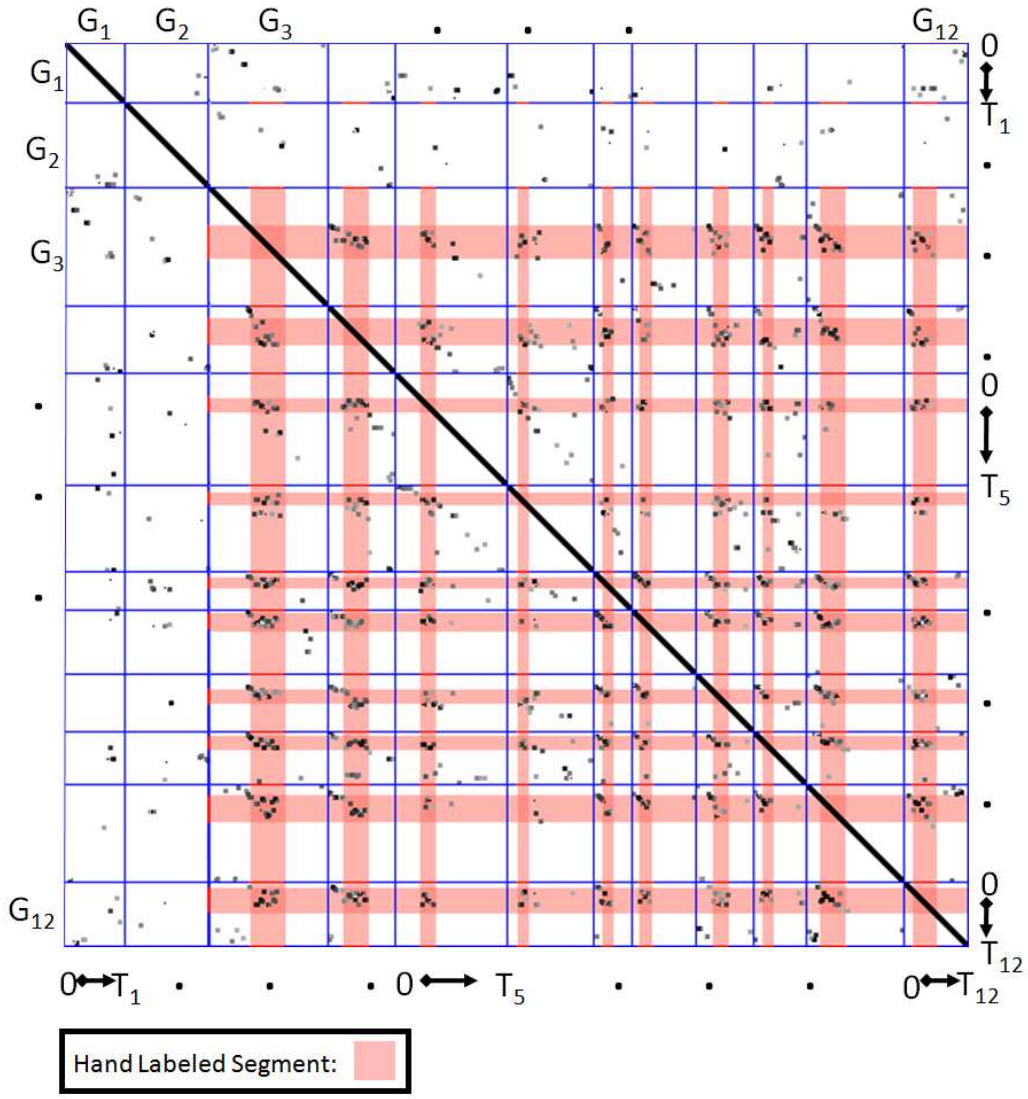


Figure 5.6: Visual representation of comparison matrices $\overline{G_{ij}}$ (Gesture Sequences G_1 and G_2 where not labeled to contain the target word as they where deemed to have translation ambiguities where the text translation contained the target word "Play" but the corresponding sign did not occur in the video)

a 1-Dimensional Gaussian filter, $\widetilde{g}(x)(\sigma = 1.5)$, to the density vector Ψ_i to create a blurred density vector $\widetilde{\Psi_i}$ =Ψ_i*$\widetilde{g}$. A video sequence is classified as a non-eligible sentence if the maximum density probability, $P(\widetilde{\Psi_i}[t_i^{Max}]|\Psi)$, falls below a set threshold. Where Ψ= $\{\Psi_1, ...,\Psi_{N_B}\}$ and t_i^{Max} is defined as the frame with the maximum density:

$$t_i^{Max} = \underset{\hat{t}}{\text{argmax}}\widetilde{\Psi_i}[\hat{t}] \tag{5.17}$$

The density probability, $P(\widetilde{\Psi_i}[t_i^{Max}]|\Psi)$, is calculated using the cumulative distribution function in Equation 5.18, where the density mean and standard deviation, $\mu(\Psi)$ and $\sigma(\Psi)$, are computed over all density sequences using Equations 5.19 - 5.21.

$$P(x|\Psi) = \frac{1}{2}\left[1 + erf\left(\frac{x - \mu(\Psi)}{\sigma(\Psi)\sqrt{2}}\right)\right] \tag{5.18}$$

$$\mu(\Psi) = \frac{1}{\ell(\Psi)}\sum_{i=0}^{N_B}\sum_{j=0}^{N_{Bi}}\Psi_i[j] \tag{5.19}$$

$$\sigma(\Psi) = \sqrt{\frac{1}{\ell(\Psi)}\sum_{i=0}^{N_B}\sum_{j=0}^{N_{Bi}}(\Psi_i[j] - \mu(\Psi))^2} \tag{5.20}$$

$$\ell(\Psi) = \sum_{i=0}^{N_B} N_{Bi} \tag{5.21}$$

From our experiments, we observed that a threshold of value of 0.55 performed best when tresholding the maximum frame density. Any sequence

which is classified as a non-eligible sentence of the target sign is not considered for the remaining automatic training steps for the current target word.

Positive sequences are sequences which have a corresponding maximum density probability which exceeds the threshold. We process all positive sequences using our density labeling algorithm which we now describe. We define the function $L_{Min}(X,t)$ to be the index of the local minima of distribution X at position t. We then find the local density minima as defined in Equations 5.22 and 5.23.

$$t^{MinS} = L_{Min}(\widetilde{\Psi_i}, t_i^{Max} - 1) \tag{5.22}$$

$$t^{MinE} = L_{Min}(\widetilde{\Psi_i}, t_i^{Max} + 1) \tag{5.23}$$

The indices of the local minima are then used to calculate a modified density vector as follows:

$$\widetilde{\Psi_i}^* = \Psi_i - \left(\frac{\widetilde{\Psi_i}[t^{MinS}] + \widetilde{\Psi_i}[t^{MinE}]}{2} \right) \tag{5.24}$$

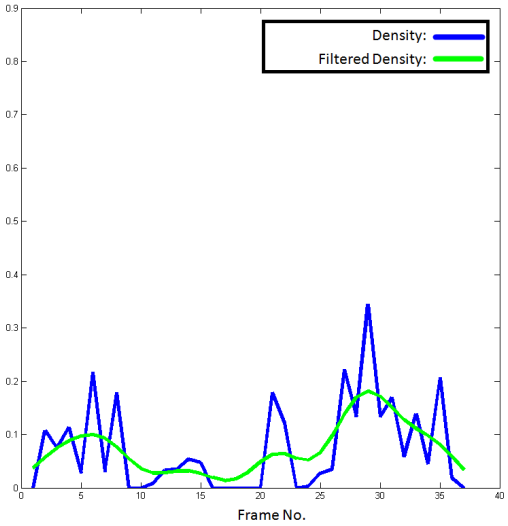
In order to identify the subsequence of the density vector $\widetilde{\Psi_i}^*$, which corresponds to the sequence of the video which the target sign is performed, we find the Maximum Sum Contiguous Subsequence (MSCS) of $\widetilde{\Psi_i}^*$. The MSCS of $\widetilde{\Psi_i}^*$ corresponds to the subsequence with the maximum value of $\sum_t^T \widetilde{\Psi_i}^*[t]$. The start and end frames, t_i^s and t_i^e, of the target word within bag i are then defined as the indices which the MSCS begins and ends. The temporal gesture and hand posture sequences which correspond to the target word are then defined as $\widehat{G_i}$= $\{G_i[t_i^s], ...G_i[t_i^e]\}$ and $\widehat{H_i}$= $\{H_i[t_i^s], ...H_i[t_i^e]\}$

respectively.

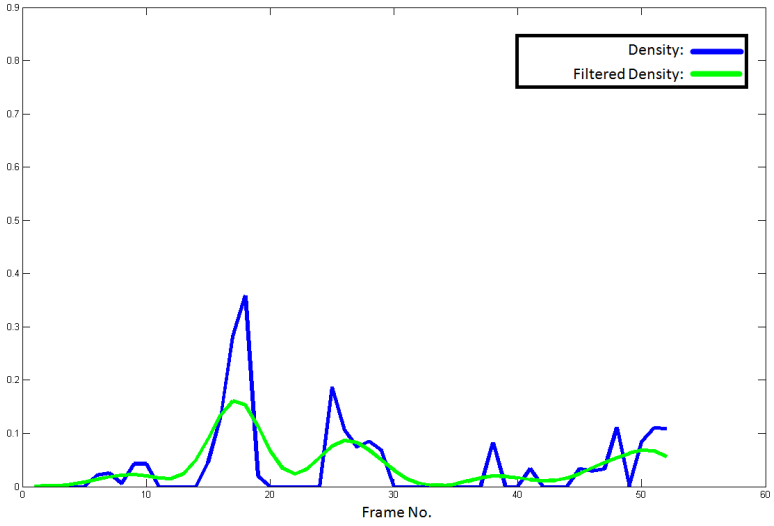
Figure 5.7 illustrates the density, and blurred density, of the target sign "Play" in 12 different sequences as well as showing the frames which were hand labeled as containing the target sign and the frames which our automatic sign labeling technique labeled. The densities of the 12 sequences shown in Figure 5.7 were calculated using the same temporal gesture density matrices shown in Figure 5.6 along with the hand shape density matrices. Figures 5.7(a) and 5.7(b) show density vectors computed from non-eligible sentences which contained translation ambiguities. It can be seen that the density of both these vectors have much smaller peaks than that of the density vectors for the remaining sentences in Figures 5.7(c) - 5.7(l) and thus where correctly labeled as non-eligible by our system.

5.5 Training And Classification

For a given sign we recognise both temporal gestures and hand postures independently and then combine the recognition results to make an overall sign classification. Our temporal gesture recognition system is based on the GT-HMM framework proposed in Chapter 4, which models the spatiotemporal pattern of the hand movements over time. The hand posture recognition framework is based on the SVM framework proposed in Chapter 3, which is automatically trained to classify key hand postures within signs.

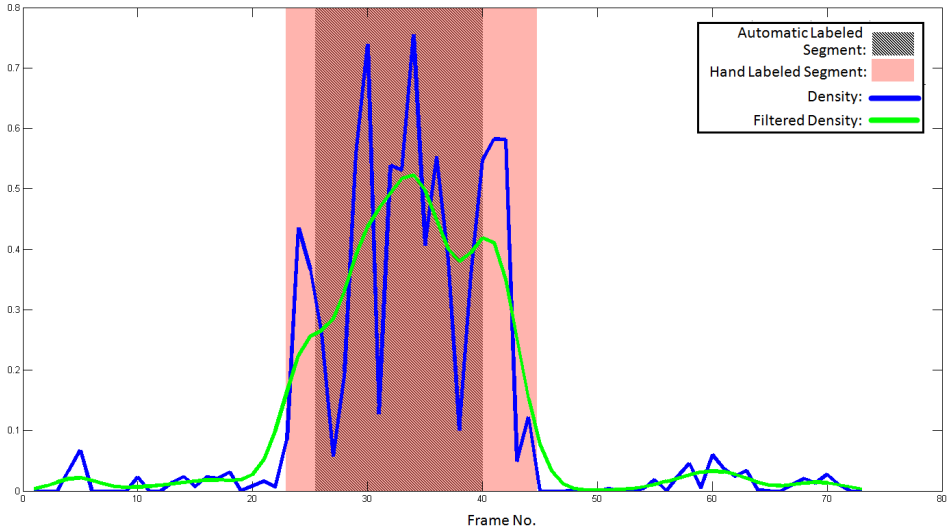


(a) Ψ_1, Correctly Labeled as Non-Eligible Sentence

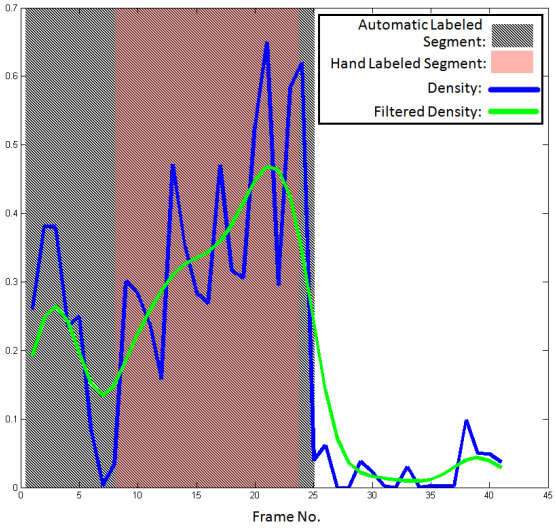


(b) Ψ_2, Correctly Labeled as Non-Eligible Sentence

Figure 5.7: Vector $\widetilde{\Psi_i}$ and Automatically Labeled target signs "Play"

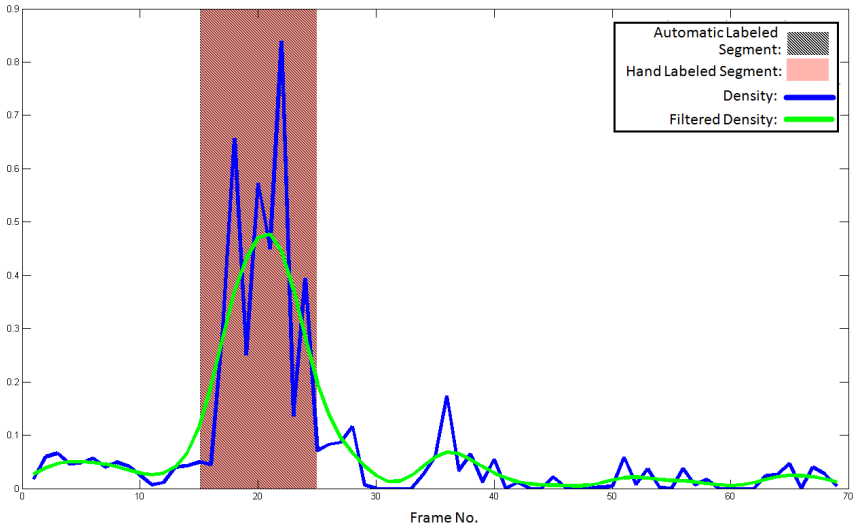


(c) Ψ_3

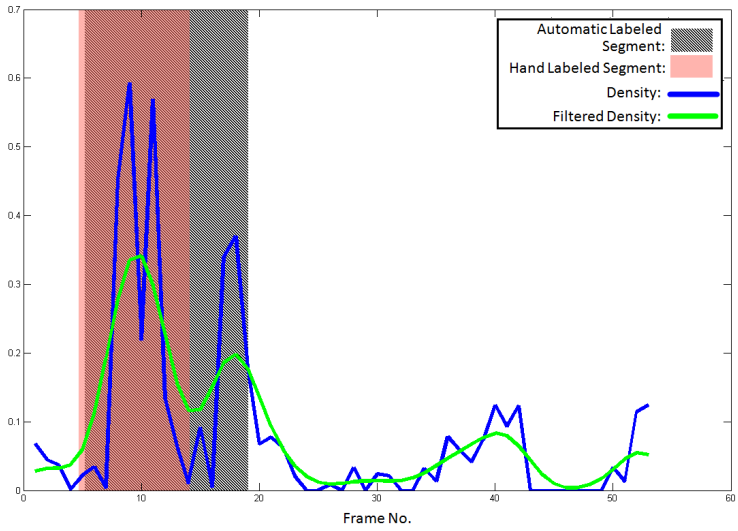


(d) Ψ_4

Figure 5.7: (cont.) Vector $\widetilde{\Psi_i}$ and Automatically Labeled target signs "Play"

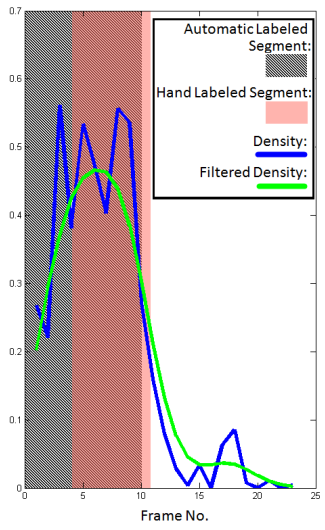


(e) Ψ_5

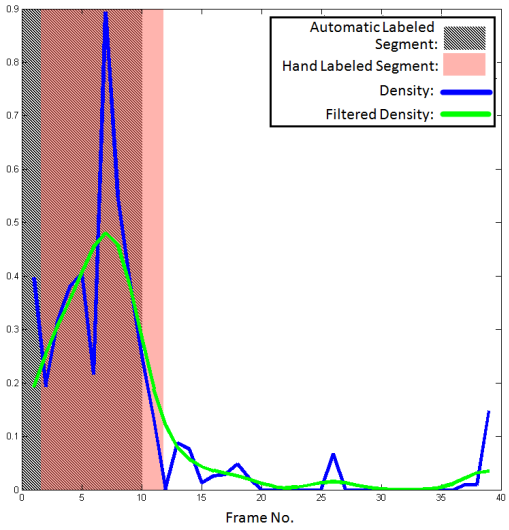


(f) Ψ_6

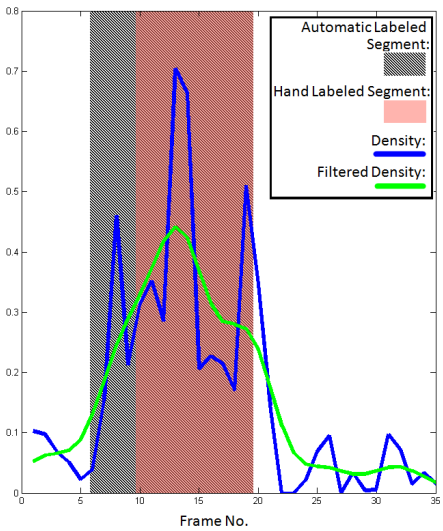
Figure 5.7: (cont.) Vector $\widetilde{\Psi_i}$ and Automatically Labeled target signs "Play"



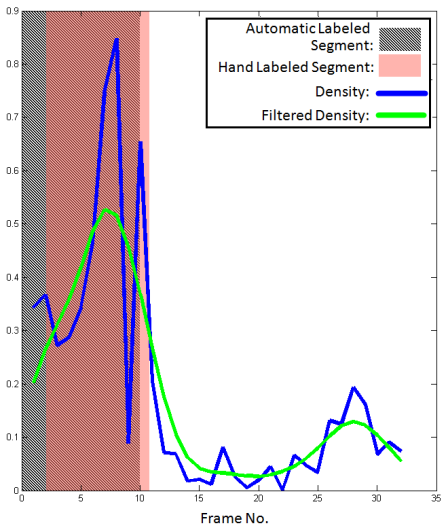
(g) Ψ_7



(h) Ψ_8

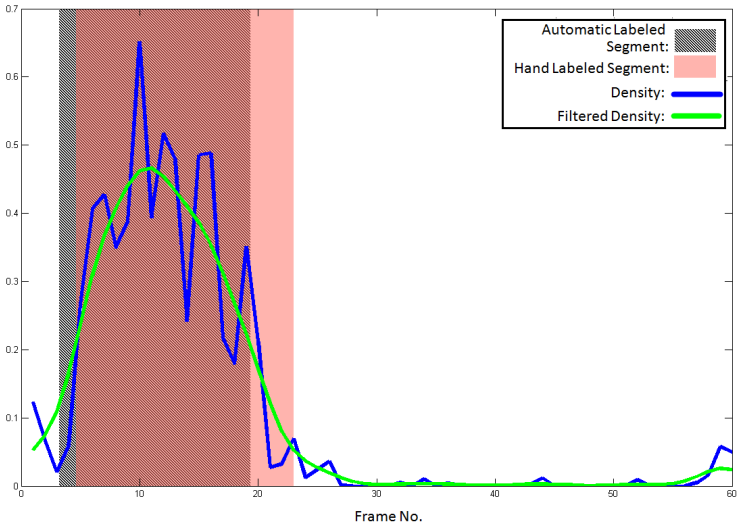


(i) Ψ_9

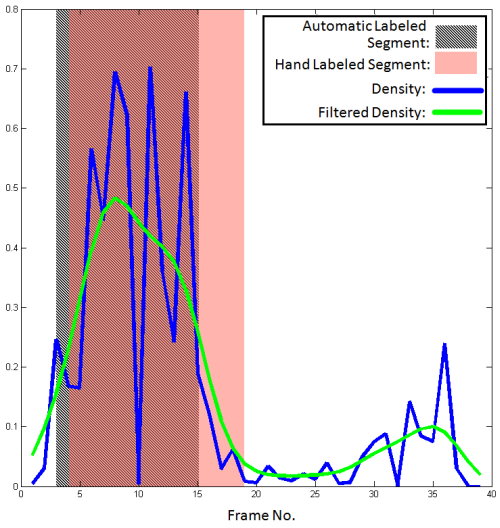


(j) Ψ_{10}

Figure 5.7: (cont.) Vector $\widetilde{\Psi_i}$ and Automatically Labeled target signs "Play"



(k) Ψ_{11}



(l) Ψ_{12}

Figure 5.7: (cont.) Vector $\widetilde{\Psi_i}$ and Automatically Labeled target signs "Play"

5.5.1 Temporal Gesture Training

Each parallel GT-HMM is trained on data from all temporal gesture subsequences $\widehat{G_k}^{Lw}$ and $\widehat{G_k}^{Rw}$ $(1 \leq k \leq K)$ extracted using our automatic sign labeling technique. Where k is the index of the kth training example for target word w, K is the total number of training examples for target word w and $\widehat{G_k}^{Lw}$ and $\widehat{G_k}^{Rw}$ are the left and right hand observations sequences respectively. The parallel GT-HMM framework is then trained using the automated gesture subunit initialisation and training technique discussed in Chapter 4.

A parallel HMM threshold model, $\overline{\lambda}' = \{\overline{\lambda}_L, \overline{\lambda}_R\}$ is then created using the network of trained parallel HMMs λ_w $(w \in W)$.

5.5.2 Hand Posture Training

While the temporal gesture recognition framework can be directly trained on data extracted from our automatic sign extraction framework described in Sections 5.4 and 5.4.2, the hand posture recognition system must apply further processing to the extracted hand posture subsequences $\widehat{H_i}$. This additional extraction process is required due to the variation in possible hand postures which can occur within a particular sign sequence. For a particular sign, there are usually only a small number of frames in which key hand postures are performed. The remaining hand postures performed in a sign are transitional postures which do not contribute to identifying the meaning of the sign.

Hand Posture Clustering

The goal of our hand posture clustering process is, given a set of hand posture subsequences $\widehat{H} = \{\widehat{H}_1, ..., \widehat{H}_{N_B}\}$, to automatically extract clusters that contain the hand postures which best represent the key postures of that sign. We now describe the key posture clustering algorithm.

We define the hand posture density subsequence as $\Psi(\widehat{H_i}) = \{\Psi(H_i)[t_i^s], ...\Psi(H_i)[t_i^e]\}$.

The probability, $P\big(\Psi(\widehat{H_i})[t]|\Psi(H^+)\big)$, of frame t of the ith bag being a key hand posture is then calculated using the cumulative distribution function defined in Equation 5.18, where $\Psi(H^+) = \{\Psi(H_1), ..., \Psi(H_{N_B})\}$.

When training an automatic sign recognition framework, the system will be trained on a set of target words w ($1 \leq w \leq W$). For a target word w, we calculate the set of gesture subsequences, $\widehat{G}^w = \{\widehat{G}_1^w, ..., \widehat{G}_{N_B^w}^w\}$ and $\widehat{H}^w = \{\widehat{H}_1^w, ..., \widehat{H}_{N_B^w}^w\}$, which contain the target word w.

For each word w, we then extract hand postures $H_i^w[t]$ which have a corresponding probability $P\big(\Psi(\widehat{H}_i^w)[t]|\Psi(H_w^+)\big) > 0.5$ and construct an initial hand posture cluster $\xi_w = \{H_i^w[1], ..., H_i^w[L(\xi_w)]\}$ where $L(\xi_w)$ total number of hand postures in ξ_w. For the initial hand posture cluster, $L(\xi_w)$ corresponds to the total number of hand postures $H_i^w[t]$ with a corresponding probability $P\big(\Psi(\widehat{H}_i^w)[t]|\Psi(H_w^+)\big) > 0.5$. Initial clusters are constructed for all words $w \in W$ resulting in an initial set of hand posture clusters $\Xi = \{\xi_1, ..., \xi_W\}$.

Using this set of initial clusters, the aim is to extract subsets of each cluster which contain the hand postures that represent key hand postures useful for discriminating between signs.

We develop an iterative validation and cluster trimming algorithm to

remove non key hand postures from each cluster. We automatically interpret the set of clusters by analysing the dissimilarity between each cluster pair using a cluster Silhouette [Rou87] representation. We define S_w, in Equations 5.25 - 5.29, as the measure of dissimilarity for the cluster ξ_w, where w is the index of the word and $L(\xi_w)$ defines the number of hand postures in the posture cluster for word w.

$$S_w = \frac{1}{L(\xi_w)} \sum_{t}^{L(\xi_w)} s_w[t] \tag{5.25}$$

$$s_w[t] = \frac{b_w[t] - a_w[t]}{max\{b_w[t], a_w[t]\}} \tag{5.26}$$

$$b_w[t] = \chi_{\widehat{l}}^{w}[t] \tag{5.27}$$

$$\widehat{l} = \underset{l:(l \neq w)}{\operatorname{argmin}} \chi_l^w[t] \tag{5.28}$$

$$a_w[t] = \chi_w^w[t] \tag{5.29}$$

Where $s_w[t]$ defines the dissimilarity for the t'th hand posture in cluster ξ_w. $\chi_l^w[t]$ defines the average distance between the t'th hand posture, in the posture cluster of word w, and all hand postures in the posture cluster of word l:

$$\chi_l^w[t] = \frac{1}{L(\xi_l)} \sum_{j=0}^{L(\xi_l)} D^H(\xi_w[t], \xi_l[j]) \tag{5.30}$$

A value of S_w close to 1 means the hand postures are appropriately clustered. If S_w is close to -1, then the hand postures are clustered poorly. Poorly clustered hand postures refer to hand postures which occur in other clusters

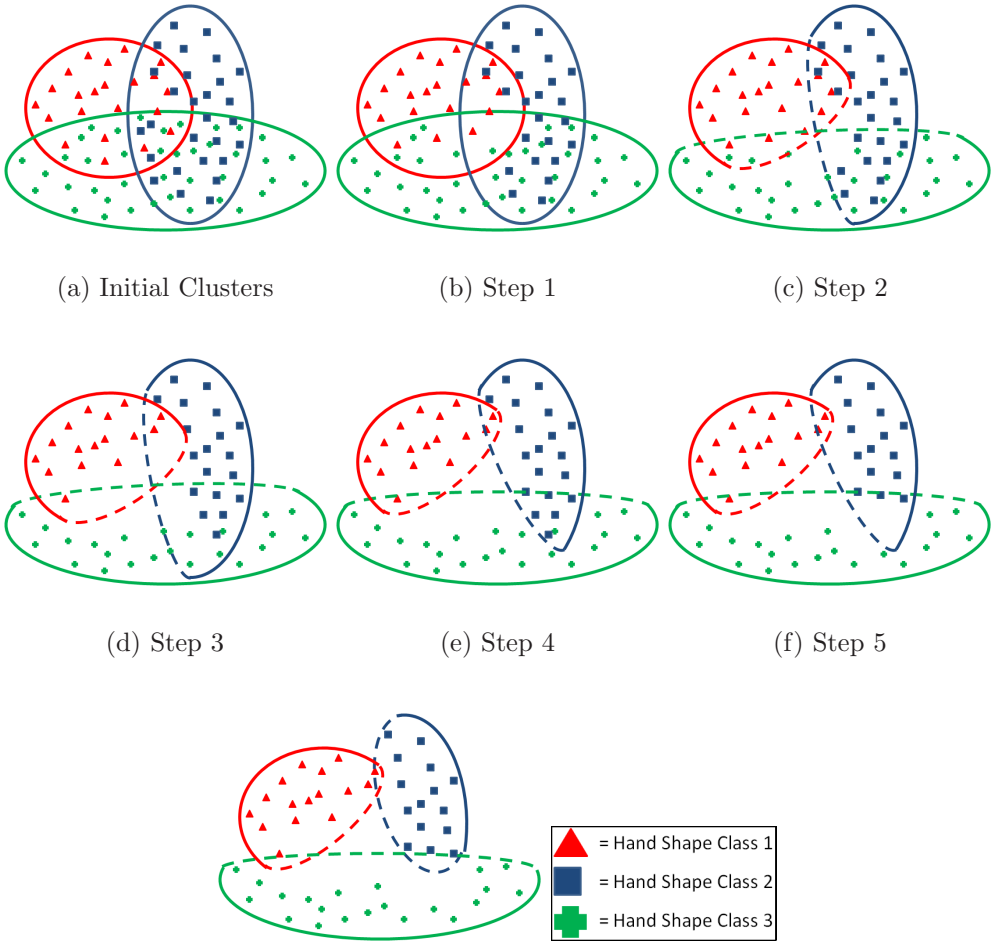
and thus are not useful for discriminating between signs. At each iteration of our validation and cluster trimming algorithm we calculate all dissimilarity measures S_w and s_w for each word w. We then remove poorly clustered hand postures based on the dissimilarity measures. If $S_w < 0$ then we perform a cluster trimming procedure were we remove hand postures $H[t]$, which have the lowest dissimilarity $s_w[t]$, from cluster ξ_w. Removing elements from cluster ξ_w in the current iteration will effect the dissimilarity values for all clusters in the next iteration, therefore to avoid over-fitting we limit the number of hand postures which can be removed, for each iteration, to a fraction of the total number of hand postures in the cluster. For the experiments we conduct we remove at most 10% of the hand postures at each iteration. This algorithm will repeat until one of two stopping criteria occurs for all hand posture clusters. The first stopping criteria for our algorithm specifies that no further postures be removed from the cluster if the dissimilarity measure $S_w > 0$. The second stopping criteria specifies that the number of postures in a cluster must not go below a predefined threshold. We set this predefined threshold to be a proportion of the total number of positive bags N_B^w for each word w. We use the heuristic that a key hand posture occurs for at least 250 milliseconds in each video, thus each key posture will appear in at least 6 frames per video. We then set the minimum number of postures, T_{min}, per cluster to be $T_{min} = N_B^w \times 6$. Algorithm 5.1 details our iterative validation and trimming procedure and Figure 5.8 shows a visualisation of

the algorithm being applied to a set of three initial hand clusters.

Input: Set of Initial Posture Clusters Ξ
Output: Set of Key Posture Clusters Ξ'

```
Converged = False
while !Converged do
    Converged = True
    foreach w ∈ W do
        Calculate Dissimilarity S_w and s_w[t] for all t's
        if S_w < 0 and L(ξ_w) > T_min then
            MaxRemove = L(ξ_w)×0.1
            Sort(s_w, ξ_w) in Increasing Order of s_w
            for j < MaxRemove do
                if s_w[j] < 0 then
                    Remove ξ_w[j] from ξ_w
                    Converged = False
                end
            end
        end
    end
end
```

Algorithm 5.1: Cluster Validation and Trimming Algorithm



(a) Initial Clusters (b) Step 1 (c) Step 2

(d) Step 3 (e) Step 4 (f) Step 5

(g) Final Step

Figure 5.8: Visualisation of Cluster Validation and Trimming Algorithm.

Hand Posture SVM Training

In Chapter 3 we propose a SVM based hand posture recognition framework. We train the hand posture SVMs on data extracted from the automatically created hand posture cluster set Ξ'.

Weighted eigenspace Size Function data and Hu moment data are then extracted from the training clusters to create the matrices $Hu_w = \{I_w[1], I_w[2], ..., I_w[L(\xi_w)]\}$ and $\zeta_w = \{\zeta_w[1], \zeta_w[2], ..., \zeta_w[L(\xi_w)]\}$ where Hu_w is the set of Hu moments and ζ_w is the set of weighted eigenspace Size Function (defined in Equation 3.7) for hand postures cluster ξ_w.

To train each Size Function SVM, SVM_w^{sf}, the matrix ζ_w is used as the positive labeled training data and $\overline{\zeta_w} := \{\zeta_k\} \forall_{k \neq w}$ is used as the negative labeled training data. Similarly, each Hu Moment SVM, SVM_w^{hu}, is trained using Hu_w as the positive labeled data and $\overline{Hu_w} := \{Hu_k\} \forall_{k \neq w}$ as the negative labeled data. The support vector machines SVM_w^{sf} and SVM_w^{hu} are then trained to maximise the hyperplane margin between their respective classes $(\zeta_w, \overline{\zeta_w})$ and $(Hu_w, \overline{Hu_w})$.

5.6 Extended Continuous Recognition

In Chapter 4 we propose a continuous spatiotemporal sign language recognition technique. We now propose an extension to the continuous spatiotemporal recognition technique which includes hand posture information.

To perform the continuous recognition, we utilise the automatically trained temporal gesture HMM framework, discussed in Section 5.5.1, and the auto-

matically trained hand posture SVM system, discussed in Section 5.5.2.

In the continuous spatiotemporal recognition algorithm, discussed in Section 4.4.2, we detect candidate end points, κ_e, and corresponding candidate start points, κ_s, from continuous streams of temporal gesture observations G= $\{\mathbf{f}_1,\mathbf{f}_2,...,\mathbf{f}_T\}$ using the automatically trained temporal gesture HMM framework. In our extended algorithm, we calculate these start and end points using the same technique as used in the spatiotemporal start and end point detection technique where start and end points are detected using spatiotemporal features only.

The start and end point detection algorithm may flag candidate gestures which overlap. Each set of overlapping candidates represent a set of gestures with similar spatiotemporal properties. In order to discriminate between gestures with similar spatiotemporal properties we incorporate the hand posture observations H= $\{C_1,C_2,...,C_T\}$. For each candidate gesture flagged, we calculate the overall probability of a particular candidate, $P(\kappa|G,H)$, by combining spatiotemporal and hand posture probabilities as described in Equations 5.31 and 5.32.

$$P(\kappa|\widehat{G},\widehat{H}) = \beta_w(\widehat{G}_{\kappa_s\kappa_e}) \times P(\alpha_w|\kappa,\widehat{H}) \quad (5.31)$$

$$P(\alpha_w|\kappa,\widehat{H}) = \max_{\kappa_s<i<\kappa_e} P(\alpha_w|I(\widehat{H}[i]),\zeta(\widehat{H}[i])) \quad (5.32)$$

Where α_w is the hand posture SVM model for word w. The hand posture probability of the candidate $P(\alpha_w|\kappa,H)$ is defined as the probability of the hand shape which best fits the gesture w. $I(H[i])$ is the set of Hu Moments

and $\zeta(H[i])$ is the weighted eigenspace Size Function computed from the hand contour $H[i]$ in frame i. Figure 5.9 shows an example of how the overall probability of a particular candidate, $P(\kappa|G,H)$, is calculated.

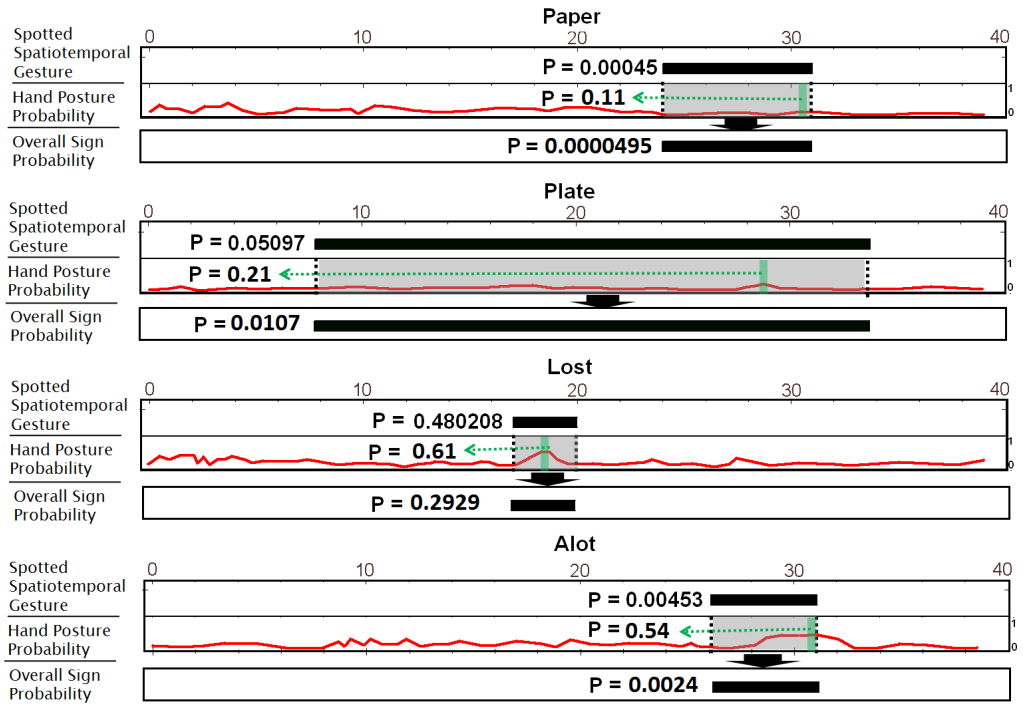


Figure 5.9: Example of computing overall sign probability by combining spatiotemporal segment probability and the best fit hand posture probability

The first step in the candidate selection algorithm is to cluster overlapping gestures with the same gesture classification. We remove all but one candidate gesture from this cluster leaving only the candidate gesture, κ^B, with the highest probability $P(\kappa|G,H)$. We repeat this step for each cluster to produce a set of candidate gestures $\Upsilon = \{\kappa^{B1}, \kappa^{B2}, ..., \kappa^{BK}\}$, where K is the total number of clusters created from grouping overlapping gestures with

the same gesture classification.

The second candidate selection step finds sets of overlapping candidates and removes the least probable candidates such that a maximum of only one candidate is detected for any given time frame. Figure 5.10 shows the time segments and gesture probabilities of the recognised gestures after the first and second candidate selection step where the signs "Lost" and "Alot" are correctly recognised from a sample sign language sentence "I Lost Alot of Books". It should be noted that the candidate gestures shown in Figure 5.10 correspond to the same set of candidates computed from combining the spatiotemporal and hand posture probabilities shown in Figure 5.9.

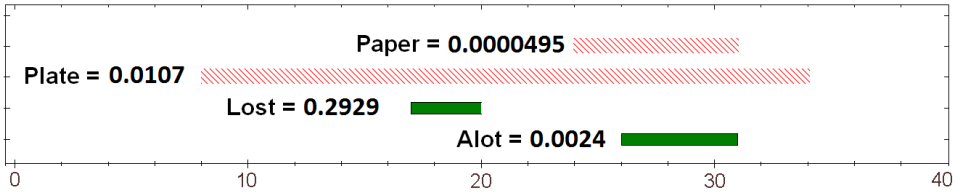


Figure 5.10: Candidate Gestures, Υ. Candidates marked in Red (Dashed) denote gestures which are removed by the second candidate selection step. Candidates in Green (Solid) denote the final recognised gestures

5.7 Experiments

5.7.1 Sign Recognition Experiments

A description of the experiments conducted to evaluate the overall performance of our automatic training and recognition system is presented in this section. We describe the data set used to evaluate our system and discuss ex-

periments carried on our automatic sign labeling technique. We then discuss the experiment carried out to evaluate the performance of our overall recognition system which combines the automatic training, hand posture classifier and temporal gesture classifier.

Data Collection

Two fluent ISL signers were recorded while they performed a total of 844 natural sign language sentences. The signers were given no instruction other than to sign to the camera and to sign sentences while trying to incorporate certain key words into the sentences. While recording the sentences a certified ISL interpreter translated each sentence through a microphone connected to the video camera. Videos were captured at 25fps with a frame size of 640 $\times$ 480.

From the set of 844 sentences, a lexicon of 30 signs was decided on based on signs which occurred frequently within the 844 sentences. The 30 key signs were then labeled within the 844 sentences. Each of the 30 signs occurred, on average, 44 times within the 844 sentences with a total of 1344 key signs occurring in the data set. The labeling process involved marking the start point and end point of each sign within each video. It is important to note that this labeling process is carried out for ground truth data only and none of these labels are used in training of the system.

Sign Labeling

The goal of the automatic sign extraction framework is to accurately detect target signs within unsegmented sentences and label them at a frame level. Due to ambiguities in the sign translation, the system must also be able to detect whether or not a target sign was actually performed in a given video even though the translation information specifies that it does occur.

When performing automatic sign extraction, we construct a set of bags B^+ which contains video sequences where the target sign is said to occur based on translation data. Using our MIL density matrix algorithm, the set of bags is then used to find similarities in the video sequences in order to label the target sign. The automatic sign labeling algorithm is then used to classify positive sentences (sentences which the target word occurs) and non-eligible sentences (sentences which the target word does not occur in the video but does occur in the text translation). For positive signs the labeling algorithm then detects start and end points of the target sign.

Since automatic sign labeling is based on a comparison of other sentences in the bag, the number of sentences and the number of possible non-eligible sentences can affect the labeling of all sentences in a bag. We first investigate the effect non-eligible sentences have on sign labeling by varying the number of videos in a bag and also varying the percentage of the bag which is made up of non-eligible sentences. We vary the number of elements in a positive bag from $5 - 20$ videos and vary the percentage of non-eligible sentences for the current target word from $0\% - 50\%$. To evaluate our automatic sign labeling system, we compare the results of the automatic labeling to that of

the ground truth data. In the automatic labeling experiments we use true positives, false negatives, true negatives and false positives to quantify the performance of the system. A true positive (TP) refers to when the classifier correctly classifies a positive sign as one that occurred in the sentence and also correctly labeled the start and end points such that the classified sign overlaps with the ground truth sign by at least 50%. A false positive (FP) refers to a sentence which is incorrectly classified as a positive sentence or a sign which the start and end points are flagged such that it does not overlap with the ground truth sign. A true negative (TN) refers to when a sentence is correctly labeled as a non-eligible sentence and a false negative (FN) refers to a sentence which is incorrectly labeled as a non-eligible sentence. For different percentages of non-eligible sentences ($0\% - 50\%$), we calculate the total precision, recall and f-measure for all 30 signs when labeled from bags which contained 5, 10, 15 and 20 sentences. Precision, recall and F-measure is defined in Equations 5.33 - 5.35.

$$Precision = \frac{\#TP}{\#TP + \#FP} \tag{5.33}$$

$$Recall = \frac{\#TP}{\#TP + \#FN} \tag{5.34}$$

$$Fmeasure = \frac{2 \times Precision \times Recal}{Precision + Recall} \tag{5.35}$$

Figure 5.11 shows the precision, recall and f-measure for different percentages of non-eligible sentences present in the set of bags B^+. Results show a

f-measure of 0.92 when there exists no non-eligible sentences, and a f-measure of 0.67 when 50% of the bag is made up of non-eligible sentences. In our data set of 844 sentences, 12.2% of the sentences contained non-eligible sentences. Thus, a good indication of how our sign labeling system would perform in reality is to evaluate the system on bags which approximately contain 12.2% non-eligible sentences. In our experiment, results show an f-measure of 0.874 when labeling signs in a bag made up of 15% non-eligible sentences. This can be interpreted as a promising result. By looking at the corresponding precision and recall values for the bag made up of 15% non-eligible sentences, we can see that our technique achieves a precision of 0.942. This means 94.2% of the data that will be used to train the spatiotemporal gesture and hand posture models is correct. The recall rate achieved was 0.833 meaning that only 16.7% of valid training data was incorrectly discarded.

An important observation to make from the results of this experiment is that, as the percentage of non-eligible sentences increases, the precision rate does not drop. A vital part of the classification of the sentences is to reduce the number of false positives since any false positives will then be used in the training of the spatiotemporal and hand posture classifiers. The consistent precision rate achieved during this experiment demonstrates that our system performs well at reducing the number of false positives.

Start and End Point Detection

The second experiment we conduct on our automatic sign extraction framework is an evaluation of the performance of the system when detecting start

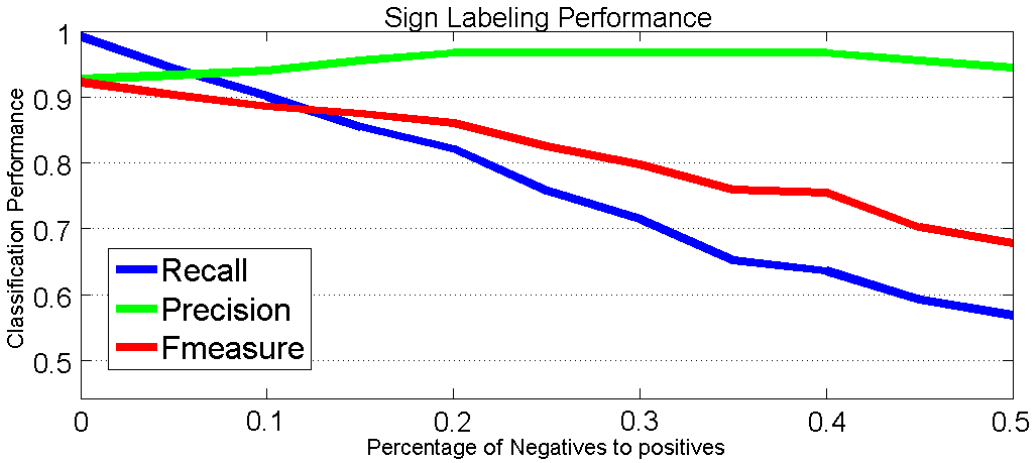


Figure 5.11: Performance of Automatic Sign Labeling and Effect of Translation Ambiguities

points and end points of positively labeled sequences. In all experiments from now on, we use a set of bags which contains 15 videos sequences of which an average of 12.2% are non-eligible sentences which contain translation ambiguities of the target word. It is important to note that although we manually control the number of non-eligible sentences in a bag during the previous experiment, in this experiment and all experiments which follow, the percentage of non-eligible sentences is controlled only by the content of our data-set and not by supervised labeling of positive and non-eligible sentences.

The sign labeling algorithm computes the density vectors for each video and automatically identifies and discards any non-eligible sentences in the set of bags. An overall f-measure of 0.88 was achieved from the automatic classification of positive and non-eligible sentences in each of the 30 bags for

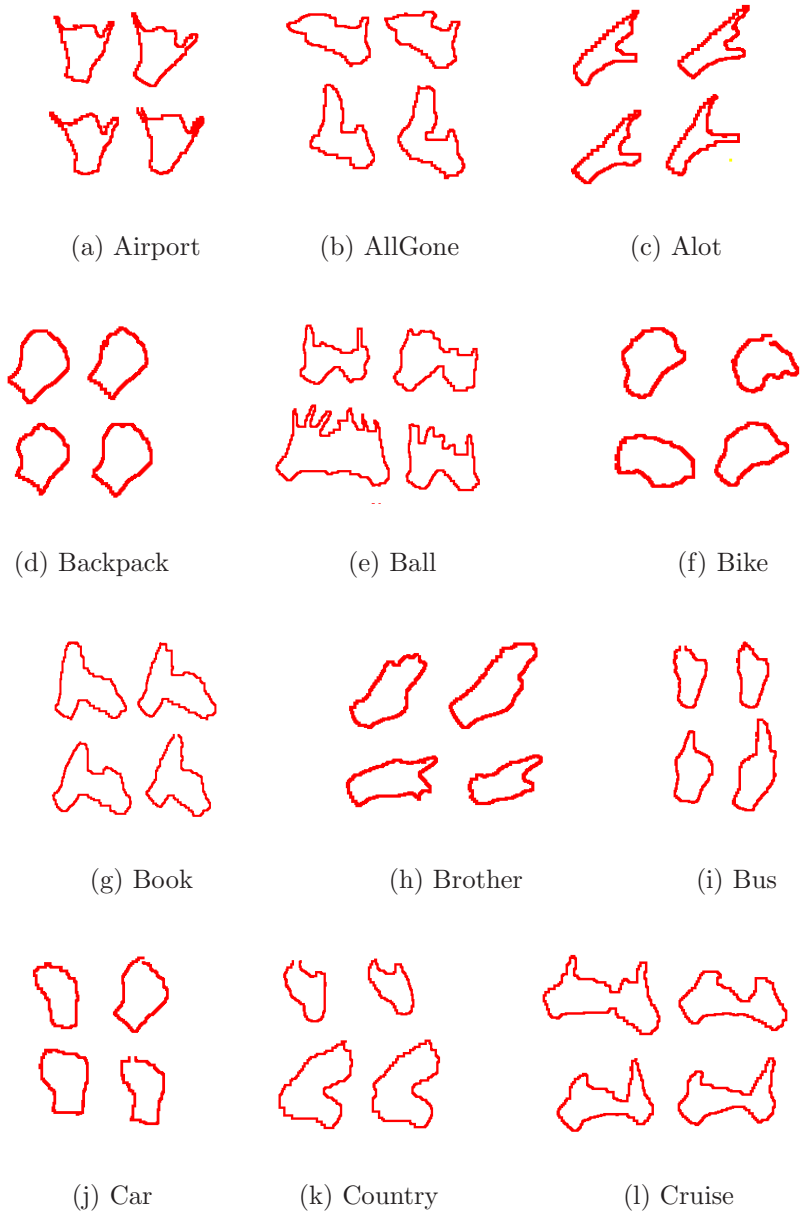
each sign. For positive videos, start and end points are then detected. To evaluate the performance of the start and end point detection, we compare the automatically detected start and end points with the ground truth start and end points and compute the mean error for each sign.

Table 5.1: Start and End Frame Error

Sign	Start Error	End Error	Sign	Start Error	End Error
Airport	±13.1	±12.9	Language	±9.2	±4.1
Allgone	±16.8	±9.3	Like	±7.3	±8.0
Alot	±10.0	±6.8	Newspaper	±6.4	±7.9
Backpack	±8.6	±6.2	Pay	±9.2	±8.3
Ball	±12.7	±14.3	Play	±3.6	±7.4
Bike	±8.8	±5.7	Rain	±3.3	±10.0
Book	±10.2	±7.4	Read	±10.7	±7.8
Brother	±8.1	±12.5	School	±10.1	±7.6
Bus	±17.5	±12.9	Shop	±12.2	±10.4
Car	±8.5	±6.6	Sister	±5.4	±6.4
Country	±11.1	±8.2	Sun	±4.7	±8.4
Cruise	±12.8	±13.1	Teacher	±8.8	±8.7
Eat	±9.0	±6.8	Understand	±8.6	±6.8
Friend	±9.25	±6.5	Warm	±5.9	±10.7
Hotel	±10.5	±9.3	Cold	±9.1	±5.3
			Mean	**±9.4**	**±8.6**

Results of the start and end point detection experiment show that our system detects the occurrence of a target word within an average of 9.4 and 8.6 frames of the ground truth data. This can be interpreted as a promising result as this result means that our technique is able to detect target sign start and end points within 0.376 and 0.344 milliseconds when compared to a human interpreter.

Following the automatic detection of start and end points of target words, our hand posture clustering algorithm is applied to the target word subsequences. Figures 5.12 - 5.14 show examples of key hand postures which were automatically extracted from these sign language subsequences.



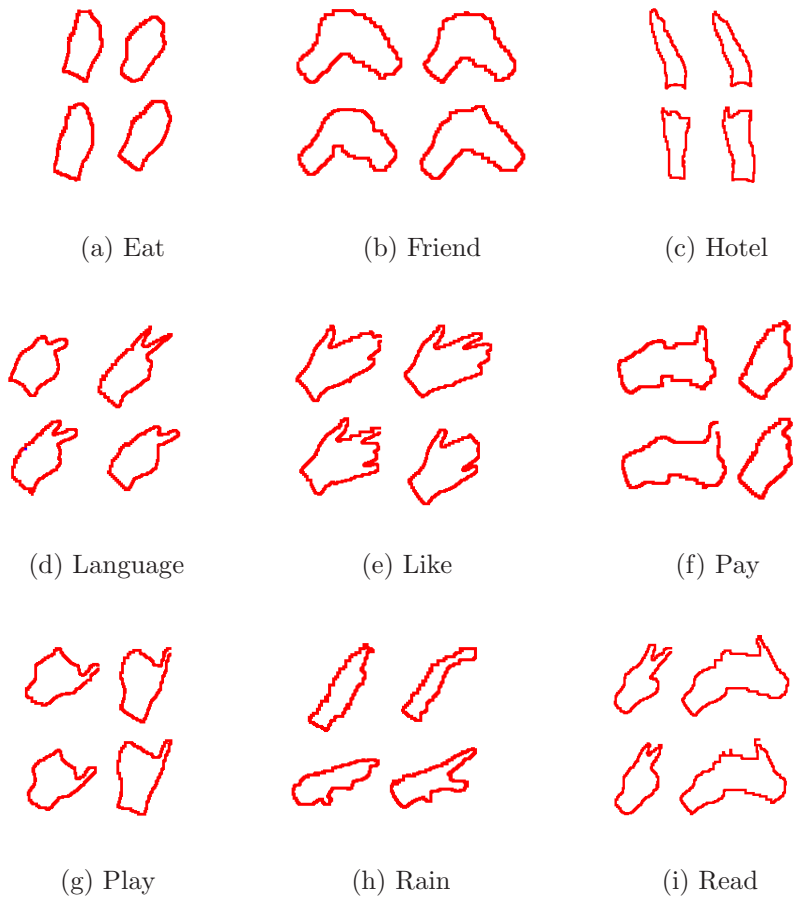
(a) Airport (b) AllGone (c) Alot

(d) Backpack (e) Ball (f) Bike

(g) Book (h) Brother (i) Bus

(j) Car (k) Country (l) Cruise

Figure 5.12: Example of automatically extracted key hand postures “Airport” - “Cruise” (4 samples hand postures shows for each sign)

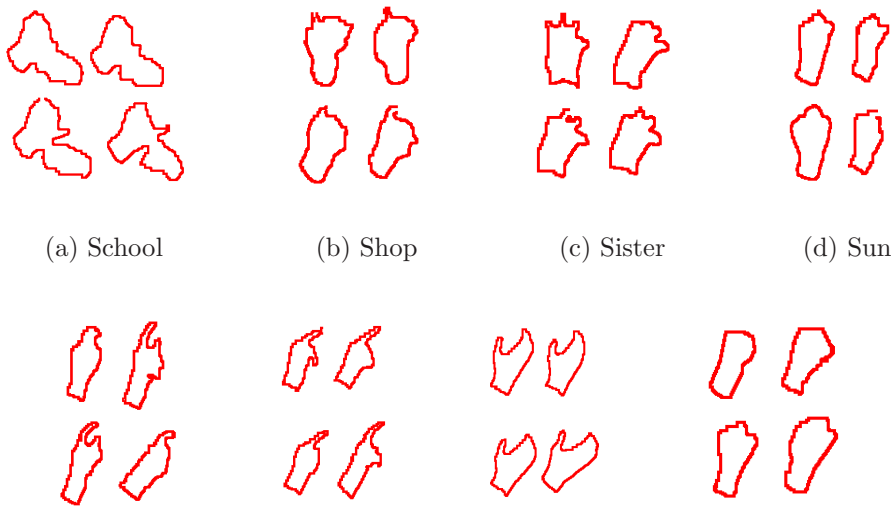


(a) Eat (b) Friend (c) Hotel

(d) Language (e) Like (f) Pay

(g) Play (h) Rain (i) Read

Figure 5.13: Example of automatically extracted key hand postures “Eat” - “Rain” (4 samples hand postures shows for each sign)



(a) School (b) Shop (c) Sister (d) Sun

(e) Teacher (f) Understand (g) Warm (h) Cold

Figure 5.14: Example of automatically extracted key hand postures "School" - "Cold" (4 samples hand postures shows for each sign)

Continuous Recognition

The overall goal of this work is to automatically train models to recognise natural sign language from unconstrained sign language sentences. We now describe experiments which were carried out to evaluate the performance of the overall sign language spotting system.

For each of the 30 target words, a set of target word subsequences were automatically extracted using the techniques we describe in Section 5.4.2. For the experiments we describe, subsequences were calculated from the same set of bags used to evaluate the start and end point labeling in Section 5.7.1. Each bag contained 15 videos sequences of which an average of 12.2% are

non-eligible sentences which contained translation ambiguities of the target word. Non-eligible sentences were automatically detected and discarded by our system with a precision of 0.931 and a recall rate of 0.856. All bags contained video sequences from only one of the two signers.

A set of parallel GT-HMMs were then automatically trained on the subsequences using the techniques we discuss in Section 4.4.2. Key hand postures for each target word were automatically extracted and a set of SVMs were then trained to recognise the key hand postures using the hand posture framework we describe in Section 5.5.2. Samples of the automatically extracted temporal subsequences and key hand postures, used in the actual training, are made available on a video as supplement to this thesis.

Given an unknown sign language sentence, we apply our recognition framework described in Section 5.6 to spot and classify signs in each sentence. To evaluate performance of our recognition framework, we test the system on the remaining set of sentences, as performed by both signers, which were not used in the training process.

In the gesture spotting and classification task, there are three types of errors: The insertion error occurs when the spotter reports a nonexistent gesture, the deletion error occurs when the spotter fails to detect a gesture, and the substitution error occurs when the spotter falsely classifies a gesture. From these error measures we define two performance metrics in Equation 5.36, where CS is the number of correctly spotted gestures, IG is the number of input gestures and IE is the number of insertion errors.

$$DetectionRatio = \frac{CS}{IG} \quad Reliability = \frac{CS}{IG + IE} \tag{5.36}$$

Table 5.2 details the performance of the recognition system when tested on the remaining sentences. Results show that the system performs well with a detection ratio of 0.832 and a reliability of 0.771. User dependent results, on Signer 1, show that the system performs with a detection ratio of 0.86 while user independent results, on Signer 2, show that the system performs well with a detection ratio of 0.764. In previous works which have used a small number of signers in the training set, results of user independent recognition evaluations have seen large decreases when compared to user dependent recognition results [OR05]. For example, in Assan et al. [AG98], accuracy for training on one signer and testing on another was 51.9% compared to 92% when the same signer supplied both training and test data. As shown in the experiments on the InteractPlay data-set in Chapter 4, user independent recognition performance using our GT-HMM threshold framework can be improved by utilising a larger number of subjects. Therefore, while the results achieved in this experiment show a decrease in the signer-independent detection rate, the fact that only one signer was represented in the training set means that a decrease of only 9.6% can be interpreted as a promising result.

We conduct a second experiment in which we evaluate the impact of the hand posture recognition component in the overall recognition of the signs. We carry out the same sign recognition experiment as before, but this time automatic key hand posture extraction was not carried out and hand shape

probability was not included in the sign recognition framework. In the absence of our hand shape techniques, results show that the overall detection rate drops by 10.8%. This shows that our automatic hand posture clustering method is an important step in developing a full weakly supervised sign training framework. It also demonstrates the importance of our hand shape classification for verification of detected spatiotemporal signs. A video showing the software implementation of our system performing sign recognition on a number of sample sentences is made available as supplement to this thesis.

The main disadvantage of our proposed framework is the computational complexity of the continuous recognition framework. During continuous experiments on the vocabulary of 30 signs, the continuous recognition system took, on average, two times the length of the video to carry out the gesture spotting algorithm. Performing the Viterbi algorithm on the HMM threshold models is the main cause of the high computation time. Reducing the number of states in the threshold model would decrease the overall computational complexity and Lee and Kim [LK99] have proposed methods to half the number of threshold model states using relative entropy.

Table 5.2: Continuous Spotter and Classifier Performance

Sign	#Correct	$\#D^{\dagger}$	$\#S^{\ddagger}$	$\#I^{\dagger\dagger}$	Det^{*}	Rel'	End Error	Start Error
rain	16	0	0	2	1	0.88	3.8	10.6
book	32	0	11	4	0.744	0.68	7.4	5.1
teacher	28	0	5	3	0.848	0.77	6	2.21
ball	22	0	5	2	0.814	0.75	3.95	2.9
shop	27	0	13	4	0.675	0.613	4.6	3.1
sun	28	0	1	3	0.965	0.875	5.3	2.71
eat	24	0	6	2	0.8	0.75	2.4	6.6
backpack	20	0	3	2	0.869	0.8	3.65	6.8
school	38	0	13	2	0.745	0.716	4.9	3.3
pay	21	0	1	3	0.954	0.84	6	2.8
read	38	0	0	4	1	0.904	3.6	17
newspaper	19	0	2	1	0.9	0.863	4	5.3
hotel	22	0	1	1	0.956	0.916	3.7	4.6
cruise	16	0	4	1	0.8	0.761	3.8	7.4
airport	26	0	2	2	0.928	0.866	2.3	8.3
car	25	0	1	2	0.96	0.892	4.8	7.9
bus	15	0	6	3	0.71	0.625	3	3.3
country	17	0	5	2	0.77	0.708	7.17	2.17
language	15	0	4	1	0.78	0.75	2.7	4.2
like	81	0	31	8	0.723	0.675	4.8	3.5
play	30	0	7	3	0.81	0.75	4.5	5
friend	35	0	10	3	0.77	0.729	2.6	4.5
brother	30	0	1	2	0.96	0.909	4.7	3.1
sister	21	0	11	2	0.65	0.617	2.9	4.9
bike	31	0	0	4	1	0.885	3	7.7
alot	23	0	3	2	0.88	0.821	2.7	1.9
allgone	16	0	3	3	0.842	0.727	2.68	3.75
understand	38	0	2	3	0.95	0.883	3.89	3.67
cold	21	0	0	1	1	0.954	4.7	4.8
warm	17	0	8	1	0.68	0.653	1.23	9.4
Total	**792**	**0**	**159**	**76**	**0.832**	**0.771**	**4.17**	**5.07**
Total Signer 1	**581**	**0**	**94**	**53**	**0.86**	**0.798**	**4.05**	**4.98**
Total Signer 2	**211**	**0**	**65**	**23**	**0.764**	**0.705**	**4.31**	**5.45**
Total No HandShape	**689**	**0**	**262**	**77**	**0.724**	**0.67**	**4.02**	**5.28**

† *Number of Deletion Errors*, ‡*Number of Insertion Errors*
††*Number of Substitution Errors*, ∗*Detection Ratio*, ′*Reliability*

5.8 Sample Application: Sign Language Teaching Environment

In this section we describe a software system developed to demonstrate a possible application of the sign language recognition technology presented in this chapter. The software is a computer vision based virtual sign language learning environment which gives real time feedback to the user on their performance of a particular sign. Signs are demonstrated by a 3D virtual teacher (see Figure 5.15) and the student is then asked to perform the same sign being performed by the teacher.

Figure 5.15: Virtual Sign Language Teacher.

The purpose of this learning environment is to provide the general community with a virtual environment for learning sign language. The virtual teaching environment provides an immersive learning experience while retaining the flexibility of learning at the users convenience on a standard PC. The system was developed with consultation and help from the Irish Deaf Society.

5.8.1 System Overview

Signs are demonstrated to the user by a 3D avatar, allowing users to view signs from any angle. Timely and appropriate feedback is a key element of our sign language teaching system and this is achieved by creating a feedback loop between the user and the software. The feedback loop is an iterative process of monitoring the users sign performance, analysing the sign and giving appropriate visual feedback to the user. The system gives feedback on the users performance of a sign by tracking hand movements and shapes through a web-cam video stream. Computer vision techniques are used to track coloured gloves which the user wears. Hand shape contours and hand movement trajectories are then extracted from the video stream and the hand posture and spatiotemporal classification models, described in Chapters 3 and 4, process the data in order to provide feedback about the performance of the sign. Figure 5.16 shows a visualisation of the architecture of the sign teaching environment and Figure 5.17 shows a screen shot of the software user interface.

5.8.2 Performance

The software described here is a system that runs in real time (approx. 16fps) on a standard laptop or PC using a standard web-cam.

In Chapters 3 and 4 we conduct a thorough evaluation of the different sign classifiers. The goal in developing this system was to investigate the feasibility of sign language recognition technology being used in real world settings by the deaf community. Therefore, in order to evaluate the feasibility

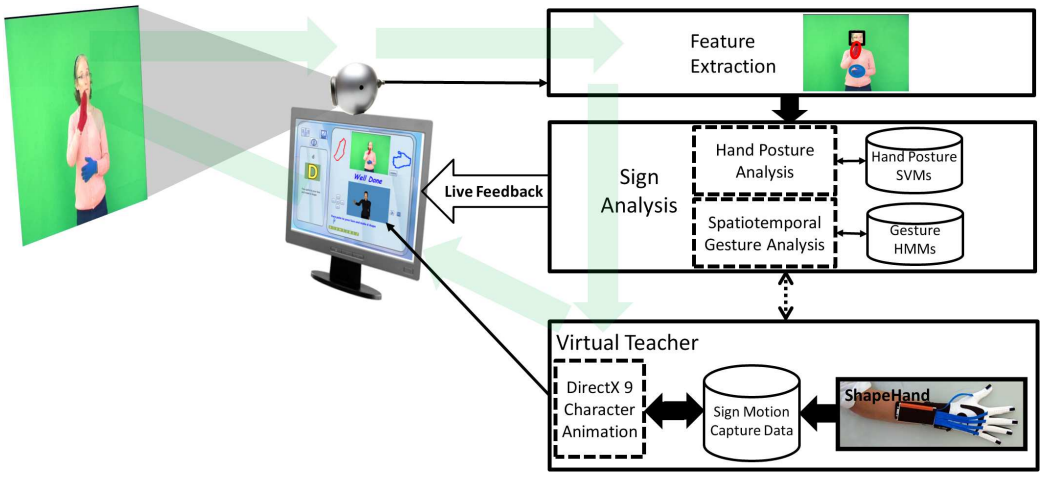


Figure 5.16: Architecture of the Sign Teaching Environment

Figure 5.17: User Interface of Sign Language Teaching Application

of this system as a sign language teaching system, we monitored five different users interacting with the software. While the five different users exhibited different levels of sign language competency, all users were able to successfully progress through an ISL alphabet lesson.

From these evaluations we conclude that our system does have the potential to be implemented as a real world sign teaching environment. Feedback, through formal and informal discussions and demonstrations of this application to the Irish Deaf Society, deaf interpreters and deaf students, indicate that technology like this could be used to greatly improve the integration of deaf people into society.

5.9 Conclusion

Previous research in sign language recognition has typically required manual labeling of sign language videos in order to extract isolated examples of particular signs to train recognition systems. In order to advance the research of sign language to the same level a speech recognition, sign vocabularies must be expanded to the same size as speech recognition vocabularies. Expanding these manually generated sign vocabularies is a very difficult, time consuming and expensive procedure [BZE09]. Therefore advancing sign language recognition research requires robust automatic training algorithms. In this Chapter we present a novel system of automatically training models for the recognition of natural sign language.

Our proposed system is capable of learning sign language from unsegmented sign language videos using the weak and noisy supervision of text

translations. Full sign language sentences are automatically segmented and isolated samples of target words are extracted from the videos. Our GT-HMM spatiotemporal gesture recognition framework is trained to recognise signs in the vocabulary and to detect movement epenthesis. Moreover, our SVM based hand posture recognition system is trained on automatically detected key hand postures using our novel eigenspace Size Function along with HU moments. The spatiotemporal and hand posture recognition systems are then combined in a continuous recognition framework to detect signs from continuous sentences.

Experiments demonstrate that our automatic sign labeling algorithm performed well when classifying positive signs and non-eligible sentences with an f-measure of 0.874 when labeling signs in a bag made up of 15% non-eligible sentences. Results also showed the sign labeling algorithm detected start and end points of target words within an average 9.4 and 8.6 frames respectively.

An evaluation of the performance of our the sign spotting system, which was automatically trained on a vocabulary of 30 signs, was carried out. The system was tested on 962 signs which occurred within continuous sentences. Results indicate that our system can detect signs from continuous sentences with a detection rate of 82.3%. The results of the experiments are very promising. These results also indicate how sensitive our recognition system is to imperfect labeling. The sign labeling experiments showed that start and end points were labeled with an average error of 9.4 and 8.6 frames respectively. With these errors present in the training set, the reconition system is still able to carry out sign recognition with a detection rate of

82.3%. These detection rates, achieved by our system, are comparable with results achieved from state of the art recognition systems trained on manual data such as the work of Yang et al [YSL09]. By way of comparison, their system was manually trained on a vocabulary of 48 signs and could detect signs with an 87% detection rate when tested on 480 signs.

The contribution of the work detailed in this chapter is that we have developed a system which can automatically learn natural signs from the weak and noisy supervision of text translations using our MIL density matrix algorithm. Another important aspect of the work detailed in this chapter is that we further demonstrate the robustness our proposed hand posture and spatiotemporal gesture models. These are important contributions to the area of natural sign language recognition as they introduce a robust framework for training a recognition system without the need for manual labeling. Freely available sign language videos, and corresponding translations, from television broadcasts can be utilised as an expansive training set which our system could be trained on. Future work will involve expanding the sign recognition vocabulary by training our system on these sign language videos available from television news broadcasts.

Chapter 6

Conclusions and Future Directions

In this thesis we detailed the development of several techniques for modeling, training and recognising natural sign language sentences. This concluding Chapter summarises our work, highlighting the contributions made, and suggests possible future directions for this work.

6.1 Contributions

As discussed in Chapter 1, there are three fundamental contributions detailed in this thesis:

- User Independent Hand Posture Measurement using the Eigenspace Size Function
- Spatiotemporal Gesture Recognition with Movement Epenthesis De-

tection

- Weakly Supervised Training using Multiple Instance Learning Density Matrices

This section summarises and comments on each of these contributions.

6.1.1 User Independent Hand Posture Measurement using the Eigenspace Size Function

An integral part of recognising hand gestures in sign language is the ability to distinguish between different hand configurations. An ideal sign recognition system should give good recognition accuracy independent of the signer performing them. A difficulty with designing user independent sign recognition systems is recognising user independent hand postures. This is due to that fact that a user independent system must cope with geometric distortions due to different hand anatomy and different performance of gestures by different people.

The first contribution discussed in this thesis is a user independent hand shape feature, a weighted eigenspace Size Function. Through experiments, based on two user independent hand posture data sets, we showed that this feature is a strong improvement on the original Size Function feature. An increase in performance of 7.4% and 6.7% was shown for our weighted eigenspace Size Function when compared to the unmodified Size Function using a simple Euclidian distance classifier.

We implemented a user independent hand posture recognition frame-

work using our weighted eigenspace Size Function and a set of Hu moments, which we showed to complement our proposed feature. A recognition framework was developed using a RBF SVM and a combination of our weighted eigenspace Size Function and Hu Moments. Results of a user independent evaluation of the recognition framework showed it achieved a ROC AUC of 0.973 and 0.935 when tested on the ISL data set and the Treisch data set respectively. We also showed that our proposed technique performs comparably with a state of the art user independent hand posture recognition method, the elastic graph matching algorithm.

The advantages of our system, in contrast to other hand posture recognition systems described in Section 2.2.3, is that our system accurately recognises hand postures independent of the person performing them and independent of the style of the posture being performed. Our system also performs this person independent recognition in real time from low resolution images of the hand taken from images where the FOV includes the full upper body. These are a significant set of advantages as the combination of these advantages makes our hand posture framework an ideal system to be used in full sign language sentence level recognition systems.

Table 6.1 gives an overview of some of the results achieved on different experiments carried out on the hand posture recognition techniques.

Table 6.1: Hand Posture Classification Results Overview

Feature	Classifier	Data Set ID	Classes	Train	Test	Performance
Size	L2 Distance	ISL	23	1	480	0.735 (AUC)
Function	L2 Distance	Triesch	20	1	48	0.756 (AUC)
Eigenspace	L2 Distance	ISL	23	1	480	0.789 (AUC)
Size Function	L2 Distance	Triesch	20	1	48	0.801 (AUC)
Weighted Eigenspace	L2 Distance	ISL	23	1	480	0.809 (AUC)
Size Function	L2 Distance	Triesch	20	1	48	0.823 (AUC)
Weighted Eigenspace	SVM	ISL	23	240	240	0.973 (AUC)
Size Function	SVM	Triesch	10	16	32	0.935 (AUC)

6.1.2 Spatiotemporal Gesture Recognition with Movement Epenthesis Detection

Recognising spatiotemporal gestures is difficult due to the inter-gesture transitions which occur between valid gestures. These inter-gesture transition periods are called movement epenthesis and are not part of either of the gestures. As such, an accurate recognition system must be able to spot meaningful gesture segments from continuous sign language sequences and identify and discard movement epenthesis.

In Section 2.3 we highlighted that there are very few works which deal with continuous sign recognition without the explicit modeling of each possible epenthesis or the use of grammar rules. The second contribution discussed in this thesis is a robust pattern recognition framework for the recognition of motion based gestures and identification of movement epenthesis without using grammar rules or explicitly modeling movement epenthesis. We implement a GT-HMM system which utilises our novel gesture subunit initialisation technique along with a HMM threshold model. Our GT-HMM models continuous multidimensional gesture observations within a parallel HMM network to recognise two-handed gestures and identify movement epenthesis from continuous sign language sentences. Moreover, we also implement a GT-HMM to recognise head and eyebrow movements used to convey non-manual signals. There are a limited number of works which deal with the combination of manual signs and non-manual signals. Another contribution of our proposed gesture recognition framework is that our model is not specific to any particular type of gesture and we demonstrated this by showing

that manual and non-manual signals can be robustly spotted and classified from within continuous sign sequences.

Experiments were conducted on isolated manual and non-manual signals in order to evaluate the performance of the HMM threshold model classifying gestures and identifying movement epenthesis. In order to compare our method to other proposed spatiotemporal gestures recognition models, evaluations where carried out on the GT-HMM, T-HMM, CRF, HCRF and LDCRF models. The LDCRF model performed best when tested on data which did not include movement epenthesis data. However, when movement epenthesis gestures where introduced to the experiment, the HMM threshold model performed best when compared to all other CRF models.

Results from continuous experiments showed that our GT-HMM framework achieved a 12.4% higher detection ratio and a 12.8% higher reliability measure than the LDCRF model when tested on 220 different hand, head and eye brow gestures. Furthermore, user independent continuous experiments showed promising results with our GT-HMM achieving a detection ratio of 0.804 when trained on sign language samples from only one signer.

Tables 6.2 and 6.3 give an overview of some of the results achieved on different experiments carried out on the spatiotemporal gesture recognition techniques.

Table 6.2: Isolated Temporal Classification Results Overview

Classifier	Data Set ID	Classes	Train	Test	Test Epenthesis	Performance
GTHMM	Hand Signs	8	10	10	0	0.977
	Hand Signs	8	10	10	100	0.976
LDCRF	Hand Signs	8	10	10	0	0.985
	Hand Signs	8	10	10	100	0.942
GTHMM	Head Movements	3	6	6	0	0.947
	Head Movements	3	6	6	25	0.936
LDCRF	Head Movements	3	6	6	0	0.952
	Head Movements	3	6	6	25	0.894
GTHMM	Eyebrow	2	5	5	0	0.951
	Eyebrow	2	5	5	20	0.948
LDCRF	Eyebrow	2	5	5	0	0.955
	Eyebrow	2	5	5	20	0.918

Table 6.3: Continuous Temporal Classification Results Overview

Classifier	Data Set	User Independent	Classes	Train	Test Instances	Performance
GTHMM	Hand Signs	No	8	10	160	0.956
GTHMM	Head Movements	No	3	6	31	0.935
GTHMM	Eyebrow Gestures	No	2	5	33	0.942
LDCRF	Hand Signs	No	8	10	160	0.831
LDCRF	Head Movements	No	3	6	31	0.903
LDCRF	Eyebrow Gestures	No	2	5	33	0.787
GTHMM	Hand Signs	Yes	8	10	160	0.804
LDCRF	Hand Signs	Yes	8	10	160	0.676

6.1.3 Weakly Supervised Training using Multiple Instance Learning Density Matrices

As discussed in Section 2.3, research on sign language recognition to date has typically required manual labeling of sign language videos in order to extract isolated examples of particular signs to train recognition systems. In order to advance the research of sign language to the same level as speech recognition, sign vocabularies must be expanded to the same size as speech recognition vocabularies. Expanding these manually generated sign vocabularies is a very difficult, time consuming and expensive procedure. Therefore, advancing sign language recognition research requires robust automatic training algorithms.

The final, and perhaps the most significant, contribution discussed in this thesis is a technique for the automatic segmentation of sign language video using the weak and noisy supervision of text translations. Moreover, the automatic training technique we propose can be utilised by our hand posture and spatiotemporal recognition models in order to learn target signs. The significance of this is that it is possible, using our proposed models, to train a full sign language recognition system with minimal human input thus allowing for the development of large vocabulary recognition systems.

Experiments demonstrate that our automatic sign labeling algorithm performed well when classifying positive signs and non-eligible sentences. Experiments also showed the sign labeling algorithm was capable of automatically detecting start and end points of target words within an average 9.4 and 8.6 frames respectively.

A final experiment was conducted in order to test the combination of all

the contributions detailed in this thesis. We evaluate our full sign recognition system which combines hand posture and spatiotemporal gesture models which were trained using our automatic training technique. The recognition system was automatically trained on a vocabulary of 30 signs. The system was then tested on 962 signs which occurred within continuous sentences. Results indicate that our system could detect signs from continuous sentences with an 83.2% detection rate.

This is the most significant result which we report in this thesis. The result achieved in this final experiment demonstrates the robustness of our hand posture and spatiotemporal gesture models. But more significantly, these models combined within our automatic training framework in order to robustly learn sign language using only the weak supervision of text translations. Using these automatically learned models, our system was then able to spot the corresponding signs in unseen sign videos.

6.2 Discussion

The combination of the three main contributions of this thesis form the basis for a potential real world sign language recognition system. While this thesis has tackled some of the big problems in the area of sign language recognition.

The hand posture recognition system proposed in this thesis can accurately, and in real time, classify hand postures independent of the person performing them. In the literature, there are very few works which attempt to build such a hand posture recognition system and the experiments which we carry out will serve as a strong benchmark for future hand posture recogni-

tion systems. In the experiments we carry out, hand posture data is extracted from images using controlled settings where the signer wither is wearing a colored glove or the background is uniform and the hand region is already known. An interesting follow on to our experiments would be to conduct similar experiments using hand posture data extracted from images with less controlled conditions.

The proposed spatiotemporal recognition framework aims to provide a framework in which temporal based gestures can be robustly learned and modeled. As well as being able to model the signs, systems must also be able to identify movement epenthesis without explicit modeling of the epenthesis themselves. In the literature there are very few works which deal with epenthesis identification without explicit modeling. We conduct a number of experiments on different data and classifier combinations in order to find a classifier which can robustly recognize key signs while also being able to identify epenthesis. We conclude from the many experiments that out proposed variation of threshold model HMMs can carry out sign classification and epenthesis identification in a more robust way than any of the other classification techniques which we experiment on. One drawback of our technique is that as the vocabulary grows, the computational complexity of continuous recognition an cause the recognition system to be unusable in a real world setting. We note that there are ways which the computational complexity of continuous recognition could be decreased such as reducing the number of states in the threshold model or building a single model with all sign HMMs and threshold model. Carry out these changes should increase computation

speed of the Viterbi algorithm. Another option may be to investigate the use of parallelizing the implementation on a GPU. Since each sign probability is calculated independently, a parallel system may provide the best improvement in computation time.

The weakly supervised training methods we propose provide a framework in which the hand posture models and spatiotemporal gesture models can be trained with minimal human input. If sign recognition vocabularies are to grow to the same size as speech recognition systems, then large corpora of signs will be needed to populate the sign database. If there are no automatic means to train these systems on then large amount of human labeling will be required for every new sign added to the data set. The method we propose uses the already existing weak labeling of sign translations associated with sign videos. Our method can pick target words from the translations and then find key features from all videos which contain that word. These key features can then be use to train the hand posture and spatiotemporal recognition models. Separate to sign language, we also envisage that this automatic training technique could have applications in other areas. It could be extended to find similarities in other large data sets. In order to conduct automatic labeling of other data sets, the only change required to our technique would be to define a similarity function relevant to the new data set.

In order to evaluate the overall combination of our three contributions a final experiment was carried out were the hand posture and spatiotemporal models were trained automatically. We concluded from this experiment that

by automatically training the hand posture and spatiotemporal models, we can still robustly recognize signs in unseen videos. This is a significant result as there exists no other works which have reported this type of experiment. Some works have proposed different variations of automatic labeling, but none of these methods have been evaluated in terms of how well the labeling performs when training the recognition models.

6.3 Future Work

There are a number of possible future directions for the work we discuss in this thesis. In this section we discuss some of the possible future directions and extensions to our work.

6.3.1 Real World Corpus

The next step in creating a real world sign language recognition system is to train the system on a real world sign language corpus. In this thesis we presented a framework for the automatic training of sign language classification models using the weak supervision of text translations. Thus, the system can be trained on real world sign language corpora that already exist by utilising sign language sentences and text translations from news for the deaf television broadcasts (e.g. RTE News for the Deaf).

This can be achieved by implementing computer vision hand and face tracking algorithms to extract features from the news videos of the signer. The extracted features, along with the video subtitles, can then be utilised

by our automatic sign extraction and training framework to train the sign classification models.

6.3.2 Natural Gesture Interfaces

Until recently, the mouse, keyboard and other button based inputs (i.e. joysticks) have been the primary input devices for computers. These devices are excellent for the purposes they were designed for. However, the application areas of computers are evolving and there is now a need for more natural input mechanisms. For example, gaming applications are evolving due to the development of new motion based controllers such as Nintendo's Wii, Sony's Playstation Move and Microsoft's Project Natal. With this in mind, new application areas outside the domain of gaming could be developed by utilising more natural input mechanisms such as gestures. This could be achieved by utilising what we know about sign language, and how the deaf learn sign language, to develop a set of gestures which could be intuitively learned and utilised to control novel applications. An example of one such application could be a ubiquitous system to aid surgeons during surgery by allowing them to control surgical robots, imaging technology and environment settings by using natural gestures.

6.3.3 Full Body Motion Analysis

Another interesting future direction for this work is in the extension of our proposed techniques for the task of general human action recognition. This could be achieved by acquiring human motion data from cameras and/or

kinematic sensors such as accelerometers, in order to interpret human poses, actions and activities.

Appendix A

Machine Learning Techniques

A.1 Hidden Markov Models

We define the elements of an HMM as follows:

- N - Number of states in the model.
- $S = \{S_1, S_2, ..., S_N\}$ - Individual states.
- q_t - State at time t.
- M - Number of distinct observation symbols per state.
- $A = \{a_{ij}\}$ - State transition probability distribution where $a_{ij} = P[q_{t+1} = S_j | q_t = S_i]$ (the probability of making a transition from state S_i to state S_j).
- $b_j(O_t)$ - The probability of a continuous observation O_t being observed in state S_j. The continuous observation probability is modeled on a

probability density function (pdf) of an M-dimensional multivariate gaussian. Where μ_j is the mean vector and Σ_j is the covariance matrix for the pdf in state j.

$$b_j(O_t) = \aleph(O_t; \mu_j, \Sigma_j) \quad \text{(A.1)}$$

$$= (2\pi)^{-\frac{N}{2}} |\Sigma|^{-\frac{1}{2}} exp\left(-\frac{1}{2}(O_t - \mu_j)^T \Sigma_j^{-1} (O_t - \mu_j)\right) \quad \text{(A.2)}$$

- $B = [\{\mu_1, \Sigma_1\}, \{\mu_2, \Sigma_2\}, ..., \{\mu_N, \Sigma_N\}]$ - Set of mean vectors and covariance matrices used to model the observation pdf for each state.
- $\pi = \{\pi_i\}$ - The initial state distribution where $\pi_i = P[q_1 = S_i]$.
- $O = O_1 O_2 ... O_T$ - Observation sequence where each O_t is a continuous observation vector and T is the number of observations in the sequence.

A.1.1 Forward Backward Algorithm

The probability of the observation sequence $O = O_1 O_2 ... O_T$ given the model λ (i.e. $P(O|\lambda)$) is calculated as follows:

We define the probability of the partial observation sequence $O_1 ... O_t$ and state S_t given the model λ as follows:

$$\alpha_t(i) = P(O_1 O_2 ... O_t, q_t = S_i|\lambda) \quad \text{(A.3)}$$

The partial probability, or forward variable, $\alpha_t(i)$ can then be solved inductively as follows:

1. Initialisation:

$$\alpha_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N \tag{A.4}$$

2. Induction

$$\alpha_{t+1}(j) = \left[\sum_{i=1}^{N} \alpha_t(i) a_{ij}\right] b_j(O_{t+1}), \quad 1 \leq t \leq T-1, \quad 1 \leq j \leq N \tag{A.5}$$

3. Termination

$$P(O|\lambda) = \sum_{i=1}^{N} \alpha_T(i) \tag{A.6}$$

The three steps combine to form the forward section of the forward backward algorithm where the purpose of the forward procedure is to compute the probability of an observation sequence given the model.

The backward part of the algorithm will be used later to optimise the parameters of the HMM in order to best model a given concept.

We define the backward variable $\beta_t(i)$ as the probability of the partial observation sequence from $t+1$ to the end:

$$\beta_t(i) = P(O_{t+1}O_{t+2}...O_T|q_t = S_i, \lambda) \tag{A.7}$$

Again, $\beta_t(i)$ can be solved inductively:

1. Initialisation:

$$\beta_T(i) = 1, \quad 1 \leq i \leq N \tag{A.8}$$

2. Induction

$$\beta_t(i) = \sum_{j=1}^{N} a_{ij} b_j(O_{t+1}) \beta_{t+1}(j), \quad t = T-1, T-2, ..., 1, \quad 1 \leq i \leq N. \tag{A.9}$$

A.1.2 Viterbi Algorithm

The goal of the Viterbi algorithm is, given the observation sequence $O = O_1 O_2 ... O_T$ and the model λ, compute the single best state sequence $Q = q_1 q_2 ... q_T$ which best explains the observations.

We define the highest probability along a single path, up to time t, as:

$$\delta_t(i) = \max_{q_1, q_2, ..., q_{t-1}} P[q_1 q_2 ... q_t = i, O_1 O_2 ... O_t | \lambda] \tag{A.10}$$

We can then define $\delta_{t+1}(i)$ as:

$$\delta_{t+1}(j) = [\max_i \delta_t(i) a_{ij}] b_j(O_{t+1}) \tag{A.11}$$

To find the complete state sequence, we keep track of the argument which maximised Equation A.11 for each t and j via the array $\psi_t(j)$. The algorithm for finding the best state sequence is as follows:

1. Initialisation:

$$\delta_1(i) = \pi_i b_i(O_1), \quad 1 \leq i \leq N \tag{A.12}$$

$$\psi_1(i) = 0 \tag{A.13}$$

2. Recursion:

$$\delta_t(j) = \max_{1\leq i\leq N}[\delta_{t-1}(i)a_{ij}]b_j(O_t), \quad 2 \leq t \leq T \tag{A.14}$$

$$\psi_t(j) = \operatorname*{argmax}_{1\leq i\leq N}[\delta_{t-1}(i)a_{ij}], \quad 2 \leq t \leq T \tag{A.15}$$

3. Termination:

$$P^* = \max_{1\leq i\leq N}[\delta_T(i)] \tag{A.16}$$

$$q_T^* = \operatorname*{argmax}_{1\leq i\leq N}[\delta_T(i)] \tag{A.17}$$

4. Path backtracking:

$$q_t^* = \psi_{t+1}(q_{t+1}^*), \quad t = T-1, T-2, ..., 1. \tag{A.18}$$

A.1.3 Baum-Welch Algorithm

The Baum-Welch algorithm is a re-estimation technique used to adjust the HMM parameters (A, B, π) to maximise the probability of the observation sequence given the model.

In order to describe the iterative update and improvement methods within the Baum-Welch algorithm we define the probability of being in state S_i at time t and state S_j at time $t+1$:

$$\xi_t(i,j) = P(q_t = S_i, q_{t+1} = S_j|O,\lambda) \quad (A.19)$$

$$= \frac{\alpha_t(i)a_{ij}b_j(O_{t+1})\beta_{t+1}(j)}{\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_t(i)a_{ij}b_j(O_{t+1})} \quad (A.20)$$

Since $\gamma_t(i)$ defines the probability of being in state S_i at time t, $\gamma_t(i)$ and $\xi_t(i,j)$ can be related by summing over j:

$$\gamma_t(i) = \sum_{j=1}^{N}\xi_t(i,j) \quad (A.21)$$

Summing $\gamma_t(i)$ over the time index t computes a quantity which can be interpreted as the expected number of times that state S_i is visited. Similarly, summing $\xi_t(i,j)$ over t can be interpreted as the expected number of transitions from state S_i to state S_j.

Based on the this concept of counting event occurrences, a set of re-estimation formulas for π, A and B are as follows:

$$\overline{\pi}_i = \text{Number of times in state } S_i \text{ at time } (t=1) \quad (A.22)$$

$$\overline{a}_{ij} = \frac{\sum_{t=1}^{T-1}\xi_t(i,j)}{\sum_{t=1}^{T-1}\gamma_t(i)} \quad (A.23)$$

$$\overline{\mu}_j = \frac{\sum_{t=1}^{T}\gamma_t(j)\cdot O_t}{\sum_{t=1}^{T}\gamma_t(j)} \quad (A.24)$$

$$\overline{\Sigma}_j = \frac{\sum_{t=1}^{T} \gamma_t(j) \cdot (O_t - \mu_j)(O_t - \mu_j)^T}{\sum_{t=1}^{T} \gamma_t(j)} \tag{A.25}$$

We define the current model $\lambda = (\pi, A, B)$ used to compute $\overline{\lambda} = (\overline{\pi}, \overline{A}, \overline{B})$ as defined in Equations A.22 - A.25.

Based on the above procedure, $\overline{\lambda}$ is iteratively used in place of λ in order to improve the probability of O being observed from the model until a limiting point is reached.

The re-estimation formulas can be derived directly by maximising Baum's auxiliary function over $\overline{\lambda}$:

$$Q(\lambda, \overline{\lambda}) = \sum_{Q} P(Q|O, \lambda) log[P(O, Q|\overline{\lambda})] \tag{A.26}$$

Baum et al. have proven that maximisation of $Q(\lambda, \overline{\lambda})$ leads to increased likelihood, i.e.:

$$\max_{\overline{\lambda}}[Q(\lambda, \overline{\lambda})] \Rightarrow P(O|\overline{\lambda}) \geq P(O|\lambda) \tag{A.27}$$

A.2 Support Vector Machines

SVMs are a set of supervised learning methods used in machine learning for the task of data classification.

A SVM constructs a hyperplane, between labeled data points, in order to classify data points not in the training set. Given a data set of n points $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$, $x_i \in \mathbb{R}^p$ and associated labels $Y = \{y_1, ..., y_n\}$, $y_i \in \{1, -1\}$,

we want to find the maximum-margin hyperplane which separates the points with $y_i = 1$ from points with $y_i = -1$.

Let the hyperplane be of the form:

$$\mathbf{w} \cdot \mathbf{x}_i + b = 0 \tag{A.28}$$

Where $\mathbf{w}$ is the normal vector and the parameter $\frac{b}{\|\mathbf{w}\|}$ determines the offset of the hyperplane from the origin along $\mathbf{w}$. We want to find $\mathbf{w}$ and b to maximise the margin between parallel hyperplanes that are as far apart as possible while still separating the data.

These hyperplanes are defined as follows:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq 1 \quad \forall \{i : y_i = 1\} \tag{A.29}$$

And

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 \quad \forall \{i : y_i = -1\} \tag{A.30}$$

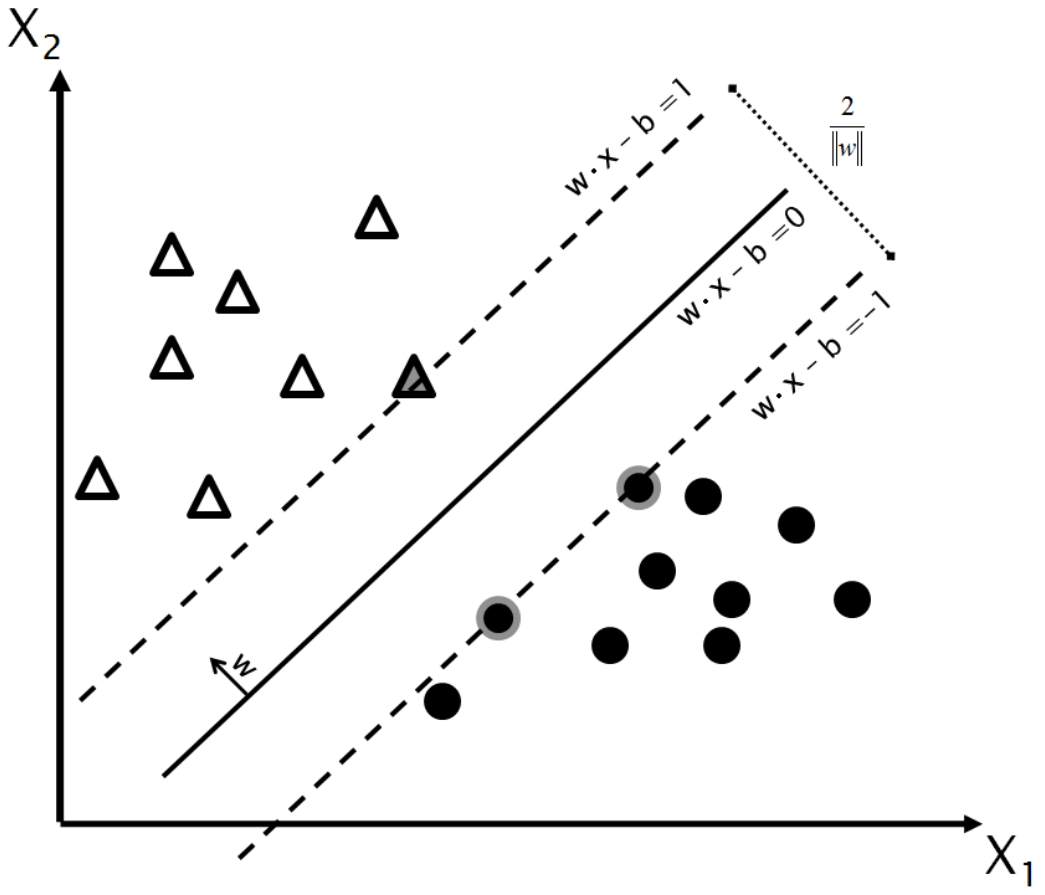


Figure A.1: Example of SVM showing maximum margin hyperplane and margins

We wish to find the two hyperplanes in a way that there are no points between them (see Figure A.1), thus we define the constraint:

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1 \quad \forall i \tag{A.31}$$

We also wish to try to maximise the distance between the two hyperplanes. The distance between the two hyperplane is $\frac{2}{\|\mathbf{w}\|}$, thus we want to

minimise $\|\mathbf{w}\|$. Minimizing this is difficult to solve as $\|\mathbf{w}\|$ involves a square root. It is possible to alter the equation by substituting $\|\mathbf{w}\|$ with $\frac{1}{2}\|\mathbf{w}\|^2$ without changing the solution.

Therefore the optimisation problem can be defined as the minimisation of $\frac{1}{2}\|\mathbf{w}\|^2$ subject to $y_i(\mathbf{w} \cdot \mathbf{x}_i + b) \geq 1(\forall i)$.

This optimisation problem can be written in terms of a Lagrangian that can be minimised with respect to $\mathbf{w}$ and b and maximised with respect to the Lagrangian multiplier α_i:

$$L(\mathbf{w}, b, \alpha) = \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{m} \alpha_i \left(y_i \left(\mathbf{w} \cdot \mathbf{x}_i + b\right) - 1\right) \tag{A.32}$$

The derivatives of the Lagrangian disappear with respect to $\mathbf{w}$ and b where:

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = 0 = \mathbf{w} - \sum_{i=1}^{m} \alpha_i y_i x_i \tag{A.33}$$

$$\frac{\partial L(\mathbf{w}, b, \alpha)}{\partial b} = 0 = \sum_{i=1}^{m} \alpha_i y_i \tag{A.34}$$

We can substitute these into A.32 to give the Wolfe Dual formulation:

$$\widetilde{L}(\mathbf{w}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \tag{A.35}$$

Support vector training therefore amounts to maximising $\widetilde{L}(\mathbf{w})$ with re-

spect to α_i subject to the constraints:

$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i \tag{A.36}$$

$$0 = \sum_{i=1}^{m} \alpha_i y_i \tag{A.37}$$

There exists a Lagrange multiplier α_i for each training point and those points which have $\alpha_i > 0$ are called support vectors and lie on one of the hyperplanes.

The decision function for an SVM is then defined as:

$$f(\mathbf{z}) = sign\left(\sum_{i=1}^{m} y_i \alpha_i (\mathbf{x}_i \cdot \mathbf{z}) + b\right) \tag{A.38}$$

Where $\mathbf{z}$ is an input and b is the bias found from:

$$b = \frac{1}{2}\left[\max_{\{i|y_i=-1\}}\left(\sum_{j=1}^{m} y_j \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j)\right) + \min_{\{i|y_i=+1\}}\left(\sum_{j=1}^{m} y_j \alpha_j (\mathbf{x}_i \cdot \mathbf{x}_j)\right)\right] \tag{A.39}$$

A.2.1 SVM Kernals

Kernals functions are used to overcome the problem of non-linearly separable data. The kernal function is used to map the data into a high dimensional feature space by providing a means of computing an inner product between two input patterns. The kernal trick allows use to compute this inner product without performing the mapping to the high dimensional space resulting in a computationally cheaper calculation compared to explicit computation of

the coordinates.

Mapping from one space to another involves a mapping from $\mathbf{x}_i \rightarrow \phi(\mathbf{x}_i)$. This involves a mapping of the inner products $\mathbf{x}_i \cdot \mathbf{x}_j \rightarrow \phi(\mathbf{x}_i) \cdot \phi(\mathbf{x}_j)$ as the feature space must be a inner product space (also known as a *pre-Hilbert* space).

A vector space is called a real inner product space if for vectors $\mathbf{u}$, $\mathbf{v}$ and $\mathbf{w}$ the following axioms hold true:

- Symmetry $(\mathbf{u} \cdot \mathbf{v}) = (\mathbf{v} \cdot \mathbf{u})$
- Additivity $((\mathbf{u} + \mathbf{v}) \cdot \mathbf{w}) = (\mathbf{u} \cdot \mathbf{w}) + (\mathbf{v} \cdot \mathbf{v})$
- Homogeneity $(\kappa\mathbf{u} \cdot \mathbf{v}) = \kappa(\mathbf{u} \cdot \mathbf{v})$
- Positivity $(\mathbf{v} \cdot \mathbf{v}) \geq 0$

An example is an inner product such as $(\mathbf{u} \cdot \mathbf{v})^2$ which can be expanded to:

$$(u_1 v_1)^2 + (u_2 v_2)^2 + 2u_1 v_1 u_2 v_2 \tag{A.40}$$

This equation can be interpreted as an inner product between two vectors in a 3-dimensional space where the feature space is related to the 2-dimensional input, $\mathbf{x}$, through the mapping:

$$\phi(\mathbf{x}) = (z_1, z_2, z_3) = (x_1^2, x_2^2, \sqrt{2}x_1 x_2) \tag{A.41}$$

There are several options available for kernals. The kernel used in this work is a RBF as defined by:

$$K(x_i, x_j) = exp(-\gamma \|x_i - x_j\|^2) \tag{A.42}$$

Combining the kernal with the objective function in Equation A.35 leads to:

$$\widetilde{L}(\mathbf{w}) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j K(\mathbf{x}_i, \mathbf{x}_j) \tag{A.43}$$

Once an optimal training solution is found through optimising the dual Wolfe Lagrangian in Equation A.43, the decision function for a new point $\mathbf{z}$ is given by:

$$f(\mathbf{z}) = sign\left(\sum_{i=1}^{m} y_i \alpha_i K(\mathbf{x}_i, \mathbf{z}) + b\right) \tag{A.44}$$

A.2.2 Probability Computation

An SVM confidence measure can be calculated for further post processing. The further a test point is from the separating hyperplane, the greater the degree of confidence should be in the classification of that point. This distance can be mapped to a probability using a method proposed by Platt [PP99]. Before the SVM decision is computed, the output is:

$$f(\mathbf{z}) = \sum_{i} y_i \alpha_i K(\mathbf{x}_i, \mathbf{z}) \tag{A.45}$$

The posterior probability $P(y = 1|f)$ can be computed using the parametric function:

$$P(y = 1|f) = \frac{1}{1 + exp(Af + B)} \tag{A.46}$$

The target classes are modified such that the classes t_i are either 0 or 1:

$$t_i = \frac{y_i + 1}{2} \tag{A.47}$$

The values for A and B can then be computed by minimizing the negative log likelihood function:

$$min \left[-\sum_i t_i log(p_i) + (1 - t_i) log(1 - p_i) \right] \tag{A.48}$$

Where p_i is calculated by:

$$p_i = \frac{1}{1 + exp(Af_i + B)} \tag{A.49}$$

Appendix B

Performance Measures

B.1 ROC Analysis

A ROC graph is a technique for organising classifiers and visualising their performance [Faw06]. ROC analysis is becoming increasingly important in the area of cost sensitive classification, classification in the presence of unbalanced classes, robust comparison of classifier performance under imprecise class distribution and misclassification costs.

Given a classifier and an instance, there are four possible outcomes:

- True Positive - when the classifier correctly returns a positive result
- True Negative - when the classifier correctly returns a negative result
- False Positive - when the classifier incorrectly returns a positive result
- False Negative - when the classifier incorrectly returns a negative result

Table B.1: Confusion Matrix

	Classifier Output	
Class Label	**p**	**n**
p	**T**rue **P**ositives	**T**rue **N**egatives
n	**F**alse **P**ositives	**T**rue **N**egatives

These results can be combined to create a confusion matrix, as shown in Table B.1.

The values in the confusion matrix are used to compile various ratios such as the True Positive Rate in Equation B.1 and the False Positive Rate in Equation B.2.

$$TruePositiveRate = \frac{TP}{TotalPositives} \tag{B.1}$$

$$TruePositiveRate = \frac{FP}{TotalNegatives} \tag{B.2}$$

For classifiers which produce probability values, representing the degree to which class the instance belongs to, setting a threshold value will determine a point in the ROC space. For instance, if probability values below or equal to a set threshold value of 0.8 are sent to the positive class, and other values are assigned to the negative class, then a confusion matrix can be calculated. Plotting the ROC point for each possible threshold value results in a curve. For example, if a probability values below or equal to a threshold value of 0.8 are sent to the positive class, and other values are assigned to the negative class, then a confusion matrix can be calculated. Plotting the ROC point for each possible threshold value results in a curve.

ROC curves are two dimensional graphs with the true positive rate plotted on the y-axis, and the false positive rate plotted on the x-axis. A ROC graph depicts the relative tradeoffs between the benefits and costs of a particular classifier. The most frequently used performance measure in ROC analysis is the AUC. The AUC of a classifier is equivalent to probability that the classifier will rank a randomly chosen positive instance higher than a randomly chosen negative instance.

Bibliography

[84799] Efficient time series matching by wavelets. In *ICDE '99: Proceedings of the 15th International Conference on Data Engineering*, page 126, Washington, DC, USA, 1999. IEEE Computer Society.

[84700] Landmarks: A new model for similarity-based pattern querying in time series databases. In *ICDE '00: Proceedings of the 16th International Conference on Data Engineering*, page 33, Washington, DC, USA, 2000. IEEE Computer Society.

[ABR64] A. Aizerman, E. M. Braverman, and L. I. Rozoner. Theoretical foundations of the potential function method in pattern recognition learning. *Automation and Remote Control*, 25:821–837, 1964.

[AG98] Marcell Assan and Kirsti Grobel. Video-based sign language recognition using hidden markov models. In *Proceedings of the International Gesture Workshop on Gesture and Sign Language*

in Human-Computer Interaction, pages 97–109, London, UK, 1998. Springer-Verlag.

[AJH01] Omar Al-Jarrah and Alaa Halawani. Recognition of gestures in arabic sign language using neuro-fuzzy systems. *Artif. Intell.*, 133(1-2):117–138, 2001.

[AKE+04] S. Askar, Y. Kondratyuk, K. Elazouzi, P. Kauff, and O. Schreer. Vision-based skin-colour segmentation of moving hands for real-time applications. In *Visual Media Production, 2004. (CVMP). 1st European Conference on*, pages 79–85, March 2004.

[AMS05] Mark. Aronoff, Irit. Meir, and Wendy Sandler. The paradox of sign language morphology. *Language*, 81, 2005.

[Bah96] B. Bahan. *Nonmanual Realisation of Agreement in American sign language*. PhD thesis, University of California, Berkely, 1996.

[BC01] Olivier Bernier and Daniel Collobert. Head and hands 3d tracking in real time by the em algorithm. In *RATFG-RTS '01: Proceedings of the IEEE ICCV Workshop on Recognition, Analysis, and Tracking of Faces and Gestures in Real-Time Systems (RATFG-RTS'01)*, page 75, Washington, DC, USA, 2001. IEEE Computer Society.

[BF95] Y. Bengio and P. Frasconi. An input/output hmm architecture. *Advances in Neural Information Processing Systems*, pages 427–434, 1995.

[BH00] Britta Bauer and Hermann Hienz. Relevant features for video-based continuous sign language recognition. In *FG '00*, page 440, Washington, DC, USA, 2000. IEEE Computer Society.

[BK02] Britta Bauer and Karl-Friedrich Kraiss. Towards an automatic sign language recognition system using subunits. In *GW '01: Revised Papers from the International Gesture Workshop on Gesture and Sign Languages in Human-Computer Interaction*, pages 64–75, London, UK, 2002. Springer-Verlag.

[BPR+04] K.A. Barhate, K.S. Patwardhan, S.D. Roy, S. Chaudhuri, and S. Chaudhury. Robust shape based two hand tracker. In *Image Processing, 2004. ICIP '04. 2004 International Conference on*, volume 2, pages 1017–1020 Vol.2, Oct. 2004.

[BS86] Charlotte Baker-Shenk. Factors affecting the form of question signals in asl. *Diversity and Diachrony*, 1986.

[BSLJ03] H. Brashear, T. Starner, P. Lukowicz, and H. Junker. Using multiple sensors for mobile sign language recognition. *Seventh IEEE International Symposium on Wearable Computers, 2003. Proceedings.*, pages 45–52, 2003.

[BZE09] Patrick Buehler, Andrew Zisserman, and Mark Everingham. Learning sign language by watching tv (using weakly aligned subtitles). In *Computer Vision and Pattern Recognition Workshops, 2009. CVPR Workshops 2009. IEEE Computer Society Conference on*, pages 2961–2968, June 2009.

[Can86] J Canny. A computational approach to edge detection. *IEEE Trans. Pattern Anal. Mach. Intell.*, 8(6):679–698, 1986.

[CB07] H. Cooper and R. Bowden. Large lexicon detection of sign language. In *CVHCI07*, pages 88–97, 2007.

[CB09] H. Cooper and R. Bowden. Learning signs from subtitles: A weakly supervised approach to sign language recognition. In *IEEE Conference on Computer Vision and Pattern Recognition 2009*, pages 2568–2574, 2009.

[CK99] Dmitry Chetverikov and Yuri Khenokh. Matching for shape defect detection. In Franc Solina and Ales Leonardis, editors, *Computer Analysis of Images and Patterns*, volume 1689 of *Lecture Notes in Computer Science*, pages 82–82. Springer Berlin / Heidelberg, 1999.

[CL01] Chih-Chung Chang and Chih-Jen Lin. *LIBSVM: a library for support vector machines*, 2001. Software available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

[COR05] Sylvie C., W. Ong, and Surendra Ranganath. Automatic sign language analysis: A survey and the future beyond lexical meaning. *IEEE Trans. PAMI*, 27(6):873–891, 2005.

[CRM00] D. Comaniciu, V. Ramesh, and P. Meer. Real-time tracking of non-rigid objects using mean shift. *Computer Vision and Pattern Recognition, 2000. Proceedings. IEEE Conference on*, 2:142–149 vol.2, 2000.

[CT01] T.F. Cootes and C.J. Taylor. Statistical models of appearance for computer vision. Technical report, Wolfson Image Analysis Unit, Imaging Science and Biomedical Engineering, University of Manchester, Manchester M13 9PT, October 2001.

[CW00] Yuntao Cui and Juyang Weng. Appearance-based hand sign recognition from intensity image sequences. *CVIU*, 78(2):157–176, 2000.

[DB08] M. Donoser and H. Bischof. Real time appearance based hand tracking. In *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*, pages 1–4, Dec. 2008.

[DBP97] A. Del Bimbo and P. Pala. Visual image retrieval by elastic matching of user sketches. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(2):121 –132, feb. 1997.

[DM09] L. Ding and A.M. Martinez. Modelling and recognition of the linguistic components in american sign language. *Journal of Image and Vision Computing*, In Press(2):257–286, 2009.

[DT02] Jiang-Wen Deng and H.T. Tsui. A novel two-layer pca/mda scheme for hand posture recognition. *Pattern Recognition, 2002. Proceedings.*, 1:283–286 vol.1, 2002.

[Faw06] Tom Fawcett. An introduction to roc analysis. *Pattern Recogn. Lett.*, 27(8):861–874, 2006.

[FF06] A. Farhadi and D. Forsyth. Aligning asl for statistical translation using a discriminative word model. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1471–1476, 2006.

[FFW07] A. Farhadi, D. Forsyth, and R. White. Transfer learning in sign language. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, June 2007.

[FGZ03] Gaolin Fang, Wen Gao, and Debin Zhao. Large vocabulary sign language recognition based on hierarchical decision trees. *Proceedings of the 5th international conference on Multimodal interfaces - ICMI '03*, page 125, 2003.

[FM10] Mariusz Flasinski and Szymon Myslinski. On the use of graph parsing for recognition of isolated hand postures of polish sign language. *Pattern Recognition*, 43(6):2249 – 2264, 2010.

[GFZC04] Wen Gao, Gaolin Fang, Debin Zhao, and Yiqiang Chen. Transition movement models for large vocabulary continuous sign language recognition. *IEEE FG 2004*, pages 553–558, May 2004.

[GK06] Ruth B. Grossman and Judy Kegl. To capture a face: A novel technique for the analysis and quantification of facial expressions in american sign language. *Sign Language Studies*, 6(3):p273–305, 2006.

[GK07] Ruth Grossman and Judy Kegl. Moving faces: Categorization of dynamic facial expressions in american sign language by deaf and hearing participants. *Journal of Nonverbal Behavior*, 31(1):23–38, 2007.

[GP09] H. Gunes and M. Piccardi. Automatic temporal segment detection and affect recognition from face and body display. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 39(1):64–84, Feb. 2009.

[GS01] Xianping Ge and Padhraic Smyth. Segmental semi-markov models for endpoint detection in plasma etching. *IEEE Transactions on Semiconductor Engineering*, 2001.

[HH98] Chung-Lin Huang and Wen-Yi Huang. Sign language recognition using model-based tracking and a 3d hopfield neural network. *Mach. Vision Appl.*, 10(5-6):292–307, 1998.

[HKR93] D. P. Huttenlocher, G. A. Klanderman, and W. A. Rucklidge. Comparing images using the hausdorff distance. *IEEE Trans. Pattern Anal. Mach. Intell.*, 15(9):850–863, 1993.

[HLO05a] Eun-Jung Holden, Gareth Lee, and Robyn Owens. Australian sign language recognition. *Mach. Vision Appl.*, 16(5):312–320, 2005.

[HLO05b] Eun-Jung Holden, Gareth Lee, and Robyn Owens. Automatic recognition of colloquial australian sign language. In *Motion and Video Computing, 2005. WACV/MOTIONS '05 Volume 2. IEEE Workshop on*, volume 2, pages 183–188, Jan. 2005.

[HMM97] B. Henrik, T. Moeslund, and C. Madsen. Real-time recognition of hand alphabet gestures using principal component analysis. *In Proceedings of The 10th Scandinavian Conference on Image Analysis, 1997.*, 1997.

[HO03] E.-J. Holden and R. Owens. Recognising moving hand shapes. *Image Analysis and Processing, 2003.Proceedings*, pages 14–19, Sept. 2003.

[HR01] Eun-Jung Holden and Owens Robyn. Visual sign language recognition. *Mutli-Image Analysis*, 2032(6):270–287–278, 2001.

[Hu62] Ming-Kuei Hu. Visual pattern recognition by moment invariants. *Information Theory, IEEE Transactions on*, 8(2):179–187, Feb 1962.

[HZW99] M. Handouyahia, D. Ziou, and S. Wang. Sign language recognition using moment-based size functions. *Proc. Int'l Conf. Vision Interface*, pages 210–216, 1999.

[IC00] Intel-Corporation. Open source computer vision library: Reference manual, 2000.

[IMT+00] I. Imagawa, H. Matsuo, R. Taniguchi, D. Arita, Shan Lu, and S. Igi. Recognition of local features for camera-based sign language recognition system. *Pattern Recognition, 2000. Proceedings*, 4:849–853 vol.4, 2000.

[JALT08] Holger Junker, Oliver Amft, Paul Lukowicz, and Gerhard Tröster. Gesture spotting with body-worn inertial sensors to detect user activities. *Pattern Recogn.*, 41(6):2010–2024, 2008.

[JGY+09] Feng Jiang, Wen Gao, Hongxun Yao, Debin Zhao, and Xilin Chen. Synthetic data generation technique in signer-independent sign language recognition. *Pattern Recogn. Lett.*, 30(5):513–524, 2009.

[JK05] C.-F. Juang and Ksuan-Chun Ku. A recurrent fuzzy network for fuzzy temporal sequence processing and gesture recognition. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 35(4):646–658, Aug. 2005.

[JM09] Agns Just and Sbastien Marcel. A comparative study of two state-of-the-art sequence processing techniques for hand ges-

ture recognition. *Computer Vision and Image Understanding*, 113(4):532 – 543, 2009.

[JRM06] A. Just, Y. Rodriguez, and S. Marcel. Hand posture classification and recognition using the modified census transform. In *Automatic Face and Gesture Recognition, 2006. FGR 2006. 7th International Conference on*, pages 351–356, April 2006.

[KBOZ04] Timon Kadir, Richard Bowden, Eng Jon Ong, and Andrew Zisserman. Minimal training, large lexicon unconstrained sign language recognition. In *BMVC 2004*, page 440, Kingston, UK, 2004. IEEE Computer Society.

[KC02] Yeon-Jun Kim and Alistair Conkie. Automatic segmentation combining an hmm-based approach and spectral boundary correction. In *In ICSLP-2002*, pages 145–148, 2002.

[KCPM00] Eamonn Keogh, Kaushik Chakrabarti, Michael Pazzani, and Sharad Mehrotra. Dimensionality reduction for fast similarity search in large time series databases. *JOURNAL OF KNOWLEDGE AND INFORMATION SYSTEMS*, 3:263–286, 2000.

[KHDM98] J. Kittler, M. Hatef, R.P.W. Duin, and J. Matas. On combining classifiers. *IEEE PAMI*, 20(3):226–239, Mar 1998.

[KWRA08] Jonghwa Kim, Johannes Wagner, Matthias Rehm, and Elisabeth Andre. Bi-channel sensor fusion for automatic sign language recognition. *2008 8th IEEE International Conference on*

Automatic Face & Gesture Recognition, pages 1–6, September 2008.

[LE09] S. Liwicki and M. Everingham. Automatic recognition of fingerspelled words in british sign language. In *IEEE CVPR Workshops 2009.*, pages 50–57, June 2009.

[LK99] Hyeon K. Lee and Jin H. Kim. An hmm-based threshold model approach for gesture recognition. *IEEE PAMI*, 21(10):961–973, 1999.

[LP96] S.X. Liao and M. Pawlak. On image analysis by moments. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 18(3):254 –266, mar. 1996.

[LS05] Attila Licsr and Tams Szirnyi. User-adaptive hand gesture recognition system with interactive training. *Image and Vision Computing*, 23(12):1102 – 1114, 2005.

[MCSLN08] L. Anton-Canalis M. Castrillon-Santana, O. Deniz-Suarez and J. Lorenzo-Navarro. Performance evaluation of public domain haar detectors for face and facial feature detection. *VISAPP 2008*, 2008.

[MGW00] Jiyong Ma, Wen Gao, and Rui Wang. A parallel multistream model for integration of sign language recognition and lip motion. In *ICMI '00: Proc of the 3rd Intl Conf on Adv in Multimodal Interfaces*, pages 582–589, 2000.

[MHRS+04] R.M. McGuire, J. Hernandez-Rebollar, T. Starner, V. Henderson, H. Brashear, and D.S. Ross. Towards a one-way American Sign Language translator. *Sixth IEEE International Conference on Automatic Face and Gesture Recognition, 2004. Proceedings.*, pages 620–625, 2004.

[MLPpE98] Oded Maron, Toms Lozano-Prez, and Tom As Lozano p Erez. A framework for multiple-instance learning. In *Advances in Neural Information Processing Systems*, pages 570–576. MIT Press, 1998.

[MQD07] L.-P. Morency, A. Quattoni, and T. Darrell. Latent-dynamic discriminative models for continuous gesture recognition. In *Computer Vision and Pattern Recognition, 2007. CVPR '07. IEEE Conference on*, pages 1–8, June 2007.

[NSL09] S. Nayak, S. Sarkar, and B. Loeding. Automated extraction of signs from continuous sign language sentences using iterated conditional modes. In *CVPR09*, pages 2583–2590, 2009.

[OL07] C Oz and M Leu. Linguistic properties based on American Sign Language isolated word recognition with artificial neural networks using a sensory glove and motion tracker. *Neurocomputing*, 70(16-18):2891–2901, 2007.

[OR05] Sylvie C. W. Ong and Surendra Ranganath. Automatic sign language analysis: A survey and the future beyond lexical

meaning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 27(6):873–891, 2005.

[PP99] John C. Platt and John C. Platt. Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods. In *Advances in Large Margin Classifiers*. MIT Press, 1999.

[PR07] Kaustubh Srikrishna Patwardhan and Sumantra Dutta Roy. Hand gesture modelling and recognition involving changing shapes and trajectories, using a predictive eigentracker. *Pattern Recogn. Lett.*, 28(3):329–334, 2007.

[PSH97] V.I. Pavlovic, R. Sharma, and T.S. Huang. Visual interpretation of hand gestures for human-computer interaction: a review. *IEEE PAMI*, 19(7):677–695, Jul 1997.

[Rab89] L.R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, Feb 1989.

[Rou87] Peter Rousseeuw. Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65, November 1987.

[Ruc97] William J. Rucklidge. Efficiently locating objects using the hausdorff distance. *Int. J. Comput. Vision*, 24(3):251–270, 1997.

[SAAR07] T. Shanableh, K. Assaleh, and M. Al-Rousan. Spatio-temporal feature-extraction techniques for isolated gesture recognition in arabic sign language. *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, 37(3):641–650, June 2007.

[Sb85] Satoshi Suzuki and KeiichiA be. Topological structural analysis of digitized binary images by border following. *Computer Vision, Graphics, and Image Processing*, 30(1):32 – 46, 1985.

[SPW98] Thad Starner, Alex Pentland, and Joshua Weaver. Real-time american sign language recognition using desk and wearable computer based video. *IEEE PAMI*, 20(12):1371–1375, 1998.

[SR89] Liddell S.K. and Johnson R.E. American sign language: The phonological base. *Sign Langauge Studies*, 64(6):195–278, 1989.

[Sto05] Jr. Stokoe, William C. Sign language structure: An outline of the visual communication systems of the american deaf. *Journal of Deaf Studies and Deaf Education, v10 n1 p3-37 Win 2005*, 2005.

[Tea80] M. R. Teague. Image analysis via the general theory of moments. *Journal of the Optical Society of America (1917-1983)*, 70:920–930, August 1980.

[TSS02] Nobuhiko Tanibata, Nobutaka Shimada, and Yoshiaki Shirai. Extraction of hand features for recognition of sign language

words. In *In International Conference on Vision Interface*, pages 391–398, 2002.

[TvdM02] J. Triesch and C. von der Malsburg. Classification of hand postures against complex backgrounds using elastic graph matching. *Image and Vision Computing*, 20(13-14):937–943, December 2002.

[UV95] C. Uras and A. Verri. Sign language recognition: An application of the theory of size functions. *6th British Machine Vision Conference*, pages 711–720, 1995.

[VA01] C Vogler and D Metaxas. A. A framework for recognizing the simultaneous aspects of american sign language. *Computer Vision and Image Understanding*, 81:358–384, 2001.

[vAKK08] Ulrich von Agris, Moritz Knorr, and Karl-Friedrich Kraiss. The significance of facial features for automatic sign language recognition. In *8th IEEE International Conference on Automatic Face and Gesture Recognition (FG 2008), Amsterdam, The Netherlands, 17-19 September 2008*, pages 1–6, 2008.

[vASZK06] Ulrich von Agris, Daniel Schneider, Jorg Zieren, and Karl-Friedrich Kraiss. Rapid signer adaptation for isolated sign language recognition. In *CVPRW '06: Proceedings of the 2006 Conference on Computer Vision and Pattern Recognition Workshop*, page 159, Washington, DC, USA, 2006. IEEE Computer Society.

[vAZC+08a] Ulrich von Agris, Jörg Zieren, Ulrich Canzler, Britta Bauer, and Karl-Friedrich Kraiss. Recent developments in visual sign language recognition. *Universal Access in the Information Society*, 6(4):323–362, 2008.

[vAZC+08b] Ulrich von Agris, Jörg Zieren, Ulrich Canzler, Britta Bauer, and Karl-Friedrich Kraiss. Recent developments in visual sign language recognition. *Univers. Access Inf. Soc.*, 6(4):323–362, 2008.

[vdKCE06] Els van der Kooij, Onno Crasborn, and Wim Emmerik. Explaining prosodic body leans in sign language of the netherlands: Pragmatics required. *Journal of Pragmatics*, 38, 2006. Prosody and Pragmatics.

[VG08] Christian Vogler and Siome Goldenstein. Facial movement analysis in asl. *Universal Access in the Information Society*, 6(4):363–374, 2008.

[VJ01] Paul Viola and Michael Jones. Rapid object detection using a boosted cascade of simple features. *CVPR, IEEE*, 1:511, 2001.

[VM99] Christian Vogler and Dimitris Metaxas. Parallel hidden markov models for american sign language recognition. In *In ICCV*, pages 116–122, 1999.

[VM04] Christian Vogler and Dimitris Metaxas. Handshapes and movements: Multiple-channel ASL recognition. *Springer LNCS* -

Gesture-Based Communication in Human-Computer Interaction, pages 1–13, 2004.

[VUFF93] A. Verri, C. Uras, P. Frosini, and M. Ferri. On the use of size functions for shape analysis. *Qualitative Vision, 1993., Proceedings of IEEE Workshop on*, pages 89–96, 14 Jun 1993.

[WCZ+07] Qi Wang, Xilin Chen, Liang-Guo Zhang, Chunli Wang, and Wen Gao. Viewpoint invariant sign language recognition. *Computer Vision and Image Understanding*, 108(1-2):87–97, 2007.

[WH99] Ying Wu and Thomas S. Huang. Human hand modeling, analysis and animation in the context of hci, 1999.

[WHM99] Ying Wu, Thomas S. Huang, and N. Mathews. Vision-based gesture recognition: A review. In *Lecture Notes in Computer Science*, pages 103–115. Springer, 1999.

[WQM+06] Sy Bor Wang, A. Quattoni, L.-P. Morency, D. Demirdjian, and T. Darrell. Hidden conditional random fields for gesture recognition. In *Computer Vision and Pattern Recognition, 2006 IEEE Computer Society Conference on*, volume 2, pages 1521–1527, 2006.

[WSG02] Chunli Wang, Shiguang Shan, and Wen Gao. An approach based on phonemes to large vocabulary chinese sign language recognition. In *IEEE FG 2002*, page 411, Washington, DC, USA, 2002. IEEE Computer Society.

[YAT02] Ming Hsuan Yang, Narendra Ahuja, and Mark Tabb. Extraction of 2d motion trajectories and its application to hand gesture recognition. *IEEE PAMI.*, 24(8):1061–1074, 2002.

[YOI92] J. Yamato, J. Ohya, and K. Ishii. Recognizing human action in time-sequential images using hidden markov model. In *Computer Vision and Pattern Recognition, 1992. Proceedings CVPR '92., 1992 IEEE Computer Society Conference on*, pages 379 – 385, jun 1992.

[YSL07] R. Yang, S. Sarkar, and B. Loeding. Enhanced level building algorithm for the movement epenthesis problem in sign language recognition. In *CVPR07*, pages 1–8, 2007.

[YSL09] Hee D. Yang, Stan Sclaroff, and Seong W. Lee. Sign language spotting with a threshold model based on conditional random fields. *IEEE PAMI*, 99(1), 2009.

[YSL10] Ruiduo Yang, Sudeep Sarkar, and Barbara Loeding. Handling movement epenthesis and hand segmentation ambiguities in continuous sign language recognition using nested dynamic programming. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 32(3):462–477, March 2010.

[ZL02] Dengsheng Zhang and Guojun Lu. A comparative study of fourier descriptors for shape representation and retrieval. In *Proc. of 5th Asian Conference on Computer Vision (ACCV*, pages 646–651. Springer, 2002.

[ZL04] Dengsheng Zhang and Guojun Lu. Review of shape representation and description techniques. *Pattern Recognition*, 37(1):1 – 19, 2004.